(12)特許協力条約に基づいて公開された国際出願

556396

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2004 年12 月9 日 (09.12.2004)

PCT

(10) 国際公開番号
WO 2004/107170 A1

(51) 国際特許分類[7]: G06F 9/45

(21) 国際出願番号: PCT/JP2004/007731

(22) 国際出願日: 2004 年5 月28 日 (28.05.2004)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2003-151472 2003 年5 月28 日 (28.05.2003) JP

(71) 出願人 (米国を除く全ての指定国について): 松下電器産業株式会社 (MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD.) [JP/JP]; 〒5718501 大阪府門真市大字門真1006番地 Osaka (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 土井 繁則 (DOI, Shigenori). 青木 博司 (AOKI, Hiroshi). 今西 祐子 (IMANISHI, Yuko).

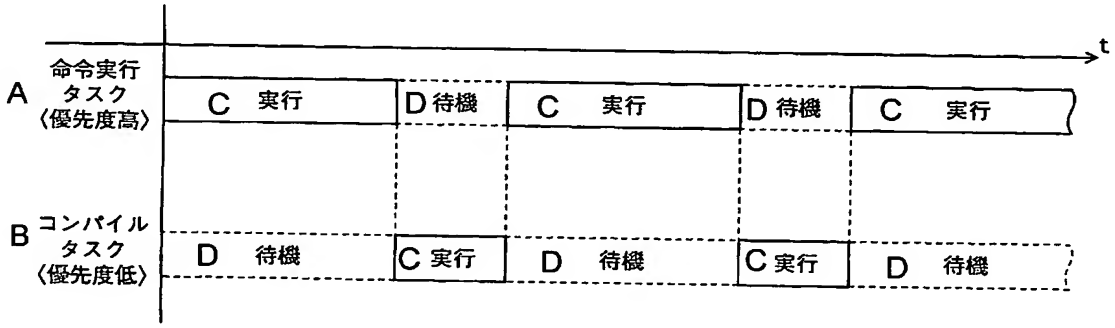
(74) 代理人: 中島 司朗 (NAKAJIMA, Shiro); 〒5310072 大阪府大阪市北区豊崎三丁目2番1号淀川5番館6F Osaka (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI,

[続葉有]

(54) Title: PROGRAM EXECUTION CONTROL DEVICE

(54) 発明の名称: プログラム実行制御装置



A...INSTRUCTION EXECUTION TASK (PRIORITY : HIGH)
B...COMPILE TASK (PRIORITY: LOW)
C...EXECUTION
D...WAITING

(57) Abstract: In a home electric appliance having a processor and the conventional JiT compiler-equipped JVM, it is possible to eliminate lowering of the execution speed caused by compiling a method when executing the method non-compiled. In order to achieve this object, when calling a method during program execution, the program execution control device judges whether the method is non-compiled. If the method is non-compiled, the byte cord of the method is executed by an interpreter processing and a request is made for compiling the method. If the method has been compiled, the native code of the method is executed. The method requested for compiling is complied by a task other than an instruction execution task such as execution of the byte cord by interpreter processing or execution of the native cord.

(57) 要約: 本発明は、従来のＪｉＴコンパイラ付ＪＶＭが実装された、プロセッサを備える家電製品において、未コンパイルのメソッド実行時に当該メソッドをコンパイルすることにより生じていた実行速度の低下を解消することを目的としてなされたものである。係る目的を達成するプログラム実行制御装置は、プログラムの実行過程においてメソッドを呼び出す際に、当該メソッドが未コンパイルであるか否かを判定し、未コンパイルであれば、当該メソッドのバイトコードをインタプリタ処理で実行すると共に当該メソッドをコンパイル

[続葉有]

WO 2004/107170 A1

NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書

する要求を行い、コンパイル済であれば、当該メソッドのネイティブコードを実行する。そして、コンパイル要求されたメソッドのコンパイルは、バイトコードのインタプリタ処理による実行又はネイティブコードの実行といった命令実行を行うタスクとは別のタスクで行われる。

明　細　書

プログラム実行制御装置、プログラム実行制御方法、制御プログラム、記録媒体

技術分野

本発明は、バイトコード列からなるプログラムをプロセッサに実行させるプログラム実行制御装置に関し、特に、プログラム実行初期段階におけるプログラムの実行速度の低下を抑制する技術に関する。

背景技術

近年、特定のプラットホームに依存することなくプログラムの実行を実現するソフトウェアである仮想マシン（Ｖｉｒｔｕａｌ　Ｍａｃｈｉｎｅ）が、組み込み機器等のプロセッサを備えた様々な家電製品に実装されている。

代表的な仮想マシンとして、Ｊａｖａ（登録商標）バイトコード（以下、単にバイトコードという。）を解釈して実行するＪＶＭ（Ｊａｖａ　Ｖｉｒｔｕａｌ　Ｍａｃｈｉｎｅ）が挙げられる。ＪＶＭの仕様については、Ｊａｖａ仮想マシン仕様　第２版（株式会社ピアソン・エデュケーション発行、ＩＳＢＮ４－８９４７１－３５６－Ｘ）に詳しく説明されている。

上述のバイトコードは、一般的にはＪａｖａ言語で記述されたソースプログラムをｊａｖａｃ等のＪａｖａコンパイラで変換することにより生成され、.ｃｌａｓｓ拡張子のクラスファイルと呼ばれるファイルに格納される。

ＪＶＭは、クラスファイルを解釈して、当該クラスファイルに含まれている各種メソッドをプロセッサに実行させるための制御を行う。メソッドとは、バイトコード列から成る特定の処理を行う手続きのことをいい、他のプログラミング言語ではメンバ関数と呼ばれているものである。

基本的にＪＶＭは、バイトコードを逐次解釈してプロセッサに処理を実行させるインタプリタとして機能するものであるが、一般的に言われているようにその実行速度は、プロセッサが直接理解できるネイティブコードに変換されたプログラムの実行速度に比べると遅い。特に、同じメソッドが繰り返し呼び出されるようなプログラムの場合、呼び出される度に同じメソッドを解釈することになるので、無駄が多い。

そこで、プログラム実行過程においてメソッドを呼び出した時に、そのメソッドが未コンパイルであるか否かを判定し、未コンパイルであれば、当該メソッドをネイティブコードに変換するＪｉＴ（Ｊｕｓｔ ｉｎ Ｔｉｍｅ）コンパイラを備えるＪＶＭが考案されている。

これにより、同じメソッドが再び呼び出されるような場合、先のメソッド呼び出し時に変換したネイティブコードを実行すればよいので、実行速度を速めることができる。

また、ネイティブコードに変換すべきメソッドを特定するためのＪｉＴコンパイラの拡張技術として、Ｊａｖａ ＨｏｔＳｐｏｔ（登録商標）と呼ばれる技術がある。これは、プログラムの実行過程において、呼び出したメソッドの呼出回数を集計し、呼出回数が閾値を超えた時に当該メソッドをネイティブコードに変換する技術である。

ＪｉＴコンパイラ付きＪＶＭのように、インタプリタ機能とコンパイラ機能を共に備えるプログラム実行装置が、特開平４－１７８７３４号公報及び特開昭６４－６２７０５号公報に開示されている。

ところで、ＪｉＴコンパイラ付きＪＶＭは、呼び出したメソッドが未コンパイルであれば直ちにコンパイルを開始し、そのコンパイル処理が終了するのを待って、生成されたネイティブコードをプロセッサに実行させるので、未コンパイルのメソッドを実行する時は、同じメソッドをインタプリタで実行するより実行速度が遅くなるという問題があった。特に、プログラムの実行開始当初では、コンパイル処理が頻発するので実行速度の低下が顕著に現れていた。

また、Ｊａｖａ ＨｏｔＳｐｏｔ技術においても、未コンパイルのメ

ソッドを呼び出した時にコンパイルを行う場合は、そのコンパイル処理が終了するのを待つので、やはり実行速度が遅くなるという問題がある。

発明の開示

本発明は、上述した問題を解決するべく、未コンパイルのメソッド実行時に当該メソッドをコンパイルすることにより生じていた実行速度の低下を解消するプログラム実行制御装置及び当該装置に関する諸技術を提供することを目的とする。

上記目的を達成する本発明に係るプログラム実行制御装置は、バイトコード列を呼び出す呼出コードを含む1又は複数のバイトコード列からなるプログラムをプロセッサに実行させるプログラム実行制御装置であって、前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定手段と、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第1手段と、呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第2手段と、前記第1手段によりなされるバイトコード列の逐次解釈実行又は前記第2手段によりなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第3手段とを備えることを特徴とする。

また、本発明に係るプログラム実行制御方法は、バイトコード列を呼び出す呼出コードを含む1又は複数のバイトコード列からなるプログラムをプロセッサに実行させるプログラム実行制御方法であって、前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに

変換済みか否かを判定する判定ステップと、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含むことを特徴とする。

また、本発明に係る制御プログラムは、バイトコード列を呼び出す呼出コードを含む１又は複数のバイトコード列からなるプログラムをプロセッサに実行させる制御プログラムであって、前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定ステップと、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含むことを特徴とする。

また、本発明に係る記録媒体は、バイトコード列を呼び出す呼出コードを含む１又は複数のバイトコード列からなるプログラムをプロセッサ

に実行させる制御プログラムを記録した記録媒体であって、前記制御プログラムは、前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定ステップと、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含むことを特徴とする。

ここで、バイトコード列は、例えば、メソッド、又はメンバ関数を指す。

これらの構成により、呼び出すバイトコード列（メソッド）が未コンパイルであれば、当該バイトコード列を逐次解釈して実行するので、従来のＪｉＴコンパイラ付ＪＶＭが未コンパイルのメソッド実行時に当該メソッドをコンパイルすることにより生じていた、プログラム実行開始当初に顕著に現れる実行速度の低下を解消することができ、また、並行して未コンパイルメソッドのコンパイルも行うので、全体的なプログラム実行速度が低下することもない。

また、前記プログラム実行制御装置は、マルチタスクオペレーティングシステムの制御下で動作し、前記第１手段によりなされる前記バイトコード列の実行又は前記第２手段によりなされる前記ネイティブコードの実行と、前記第３手段によりなされる前記変換処理の実行は、それぞれ異なるタスクとして実行され、前記第１手段及び前記第２手段のタスク実行は、前記第３手段のタスク実行より優先度が高いとしてもよい。

この構成によると、コンパイル実行タスクより命令実行タスクが優先的に実行されることになるので、プログラム実行開始当初に顕著に現れる実行速度の低下を解消することができる。

また、前記プログラム実行制御装置は、前記第１手段又は前記第２手段のタスク実行において待ち状態が発生した場合、前記第３手段のタスク実行に切り替える切替手段を更に備えるとしてもよい。

この構成によると、命令実行タスクにおいて待ち状態が発生した場合にコンパイルタスクの実行に切り替えるので、コンパイル処理を速やかに実行することができる。コンパイル処理が速やかに実行されれば、命令実行におけるインタプリタ処理の回数が減り、全体的なプログラム実行速度を速めることができる。

また、前記プログラム実行制御装置は、前記第１手段によりなされる前記要求に応じて、当該要求に係るバイトコード列をネイティブコードに変換するための情報であるコンパイル要求情報を記憶手段に登録し管理する要求管理手段を更に備え、前記第３手段は、前記第１手段によりなされるバイトコード列の逐次解釈実行又は前記第２手段によりなされる前記ネイティブコードの実行と並行して、前記要求管理手段によって前記記憶手段に登録されているコンパイル要求情報に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させるとしてもよい。

この構成によると、要求管理手段によって記憶手段に登録及び管理されているコンパイル要求情報に基づいてバイトコード列のコンパイルを実行することができる。

また、前記要求管理手段は、前記要求を先に受けたものからの順番となるキューで各コンパイル要求情報を登録して管理し、前記第３手段は、前記キューの先頭位置にあるコンパイル要求情報から順番に前記変換処理を前記プロセッサに実行させるとしてもよい。

この構成によると、コンパイル要求を受けた順番でバイトコード列のコンパイルを行うことができる。

また、前記要求管理手段は、前記第１手段によりなされる前記要求に係るバイトコード列のコンパイル要求情報が前記記憶手段に既に登録されている場合に、２重登録を行わないとしてもよい。

この構成によると、既に登録されているコンパイル要求情報を２重登録しないので、重複して同じメソッドをコンパイル処理することを防ぐことができる。

また、前記プログラム実行制御装置は、各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段を更に備え、前記要求管理手段は、前記第１手段によりなされる前記要求に応じて、取得された優先度情報を参照し、当該要求に係るバイトコード列の優先度を特定する特定手段と、前記記憶手段に登録されている各コンパイル要求情報に係るバイトコード列の優先度と、前記特定手段により特定された優先度とを比較する比較手段と、比較した結果に基づいて、優先度の高いもの順の配列となるように、前記要求に係るバイトコード列のコンパイル要求情報のキュー位置を決定する決定手段とを備えるとしてもよい。

この構成によると、優先度の高いバイトコード列から先にコンパイルすることができる。

また、前記プログラム実行制御装置は、あるバイトコード列と関連する関連バイトコード列を対応付けている対応情報を取得する対応情報取得手段と、前記対応情報を参照して、前記要求に係るバイトコード列と対応付けられている関連バイトコード列があるか否かを検索する検索手段とを更に備え、前記検索手段による検索結果、関連バイトコード列が検出された場合、前記要求管理手段は、検出された関連バイトコード列のコンパイル要求情報を記憶手段に登録するとしてもよい。

この構成によると、コンパイル要求されたバイトコード列と関連するバイトコード列のコンパイル要求情報を登録することができる。

また、前記プログラム実行制御装置は、各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段を更に備え、前記第３手段は、前記記憶手段に複数登録されているコンパイル要求情報に係るバ

イトコード列のうち、取得された前記優先度情報に基づいて優先度の高いものから順番に前記変換処理を前記プロセッサに実行させるとしてもよい。

この構成によると、優先度の高いバイトコード列から先にコンパイルすることができる。

また、前記要求管理手段は、繰り返し前記要求がなされるバイトコード列については、その要求回数を計数して、計数した要求回数を、当該バイトコード列のコンパイル要求情報に含めて前記記憶手段に記録する回数記録手段と、前記要求回数の閾値を取得する取得手段とを備え、前記第３制御手段は、前記要求回数が閾値を超えたものから順番に前記変換処理を前記プロセッサに実行させるとしてもよい。

この構成によると、閾値を超えるコンパイル要求回数となったバイトコード列からコンパイルすることができる。

また、前記要求管理手段は、繰り返し前記要求がなされるバイトコード列については、その要求回数を計数し、計数した要求回数を当該バイトコード列のコンパイル要求情報に含めて前記記憶手段に記録する回数記録手段と、記録した各コンパイル要求情報の要求回数を比較して、要求回数が多いもの順となるように各コンパイル要求情報のキュー位置を入替える入替手段とを更に備えるとしてもよい。

この構成によると、変換要求の多いバイトコード列を先にコンパイルすることができるので、命令実行におけるインタプリタ処理の回数が減り、全体的なプログラム実行速度を速めることができる。

また、前記要求管理手段は、優先度の異なる複数のキューを管理しており、前記第３手段は、優先度が最も高いキューに登録されているコンパイル要求情報に係るバイトコード列から順番に前記変換処理を前記プロセッサに実行させるとしてもよい。

この構成によると、コンパイル要求情報を登録するキューで、優先的にコンパイルすべきバイトコード列とそうでないバイトコード列を区別することができる。

また、前記プログラム実行制御装置は、前記プログラムの実行前に、ネイティブコードに変換すべき複数のバイトコード列を示す特別要求情報を取得する特別要求情報取得手段を更に備え、前記要求管理手段は、取得された特別要求情報に示される各バイトコード列のコンパイル要求情報を、優先度が最も高いキューに一括登録して管理するとしてもよい。

この構成によると、優先的にコンパイルすべきバイトコード列を特別要求情報に含めておくことで、これらのバイトコード列を優先的にコンパイルすることができる。

また、前記プログラム実行制御装置は、前記判定手段により、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定されたときに、当該バイトコード列が現在、前記変換処理の途中であるか否かを判定する第２判定手段と、前記バイトコード列が現在、前記変換処理の途中であると判定された場合に、当該変換処理が終了するのを待って、変換されたネイティブコードを前記プロセッサに実行させる第４手段とを備えるとしてもよい。

この構成によると、未コンパイルのバイトコード列の実行時に、当該バイトコード列がコンパイル中であれば、そのコンパイルが終了するのを待って、変換されたネイティブコードを実行することができる。

また、前記プログラム実行制御装置は、各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段と、前記変換処理を行うバイトコード列の優先度と前記命令実行タスクの優先度とを比較する比較手段と、比較した結果、前記バイトコード列の優先度の方が命令実行タスクの優先度より高い場合に、前記コンパイルタスクの優先度を一時的に高くする優先度変更手段とを更に備えるとしてもよい。

この構成によると、命令実行タスクの優先度より高い優先度のバイトコード列をコンパイルする場合、一時的にコンパイルタスクの優先度を高くするので、当該バイトコード列についてはコンパイルを速やかに行うことができる。

図面の簡単な説明

図１は、命令実行タスクとコンパイルタスクの実行状態推移を示す図である。

図２は、実施形態１のプログラム実行制御装置１の機能構成図である。

図３は、Ｊａｖａ言語で記述されたソースプログラムの一例を示す図である。

図４は、実施形態１のキュー記憶部１６に登録されているコンパイル要求情報の具体例を示す図である。

図５は、コンパイル済メソッド情報テーブルの一例を示す図である。

図６は、命令実行処理を説明するためのフロー図である。

図７は、実施形態２のプログラム実行制御装置１Ａの機能構成図である。

図８は、実施形態２のキュー記憶部１６Ａに登録されているコンパイル要求情報の具体例を示す図である。

図９は、コンパイル要求情報の順番入替処理を説明するためのフロー図である。

図１０は、実施形態３のプログラム実行制御装置１Ｂの機能構成図である。

図１１は、クラスファイルから取得した優先度情報テーブル２３の具体的な内容を示す図である。

図１２は、キュー記憶部１６Ｂに登録されているコンパイル要求情報の具体例を示す図である。

図１３は、コンパイル要求情報の登録位置決定処理を説明するためのフロー図である。

図１４は、実施形態４のプログラム実行制御装置１Ｃの機能構成図である。

図１５は、一括登録情報２４の具体的な内容を示す図である。

図１６は、キュー記憶部１６Ｃに登録されているコンパイル要求情報の具体例を示す図である。

図１７は、実施形態５のプログラム実行制御装置１Ｄの機能構成図である。

図１８は、関連メソッドテーブル２８の具体例を示す図である。

図１９は、関連メソッド検索処理を説明するためのフロー図である。

図２０は、実施形態６のプログラム実行制御装置１Ｅの機能構成図である。

図２１は、優先度継承処理を説明するためのフロー図である。

図２２は、命令実行処理の変形例を説明するためのフロー図である。

発明を実施するための最良の形態

＜概要＞

本発明に係るプログラム実行制御装置は、メモリ又はハードディスク等の記憶媒体と、プロセッサを備えるコンピュータ装置であり、具体的には携帯電話機や、デジタル放送受信装置等である。

プログラム実行制御装置が有する記憶媒体には、マルチタスクＯＳ（Ｏｐｅｒａｔｉｎｇ　Ｓｙｓｔｅｍ）、ＪＶＭ、クラスファイル、そして、よく使われる汎用的なクラス又はメソッド等の部品群から成るクラスライブラリ等が格納されており、本プログラム実行制御装置の各種機能は、記憶媒体に格納されているマルチタスクＯＳ、及び当該マルチタスクＯＳの制御下において動作するＪＶＭに従ってプロセッサが動作することにより実現される。

プログラム実行制御装置は、プログラムの実行過程においてメソッドを呼び出す場合、当該メソッドが未コンパイルであるか否かを判定し、未コンパイルであれば、当該メソッドのバイトコードをインタプリタ処理で実行すると共に当該メソッドをコンパイルする要求を登録し、コンパイル済であれば、当該メソッドのネイティブコードを実行する。そして、登録されたコンパイル要求に係るメソッドのコンパイルは、バイトコードのインタプリタ処理による実行又はネイティブコードの実行といった命令実行を行うタスクとは別のタスクで行う。

つまり、バイトコード又はネイティブコードの実行を行うタスクを命令実行タスク、コンパイル処理を行うタスクをコンパイルタスクと呼ぶとすると、命令実行タスクとコンパイルタスクを非同期で並行処理している点が、本発明の特徴である。

図１は、命令実行タスクとコンパイルタスクの実行状態推移を示す図である。

本プログラム実行制御装置は、プリエンプティブで優先度ベースのタスクスケジューリングによりシングルプロセッサ資源の割り当てを行っている。

各タスクの優先度は、命令実行タスクを高とし、コンパイルタスクを低としており、優先度高のタスクに割り当てられるプロセッサ時間配分と優先度低のタスクに割り当てられるプロセッサ時間配分の比率は、９：１としている。また、命令実行タスクの実行において待ち状態が発生した場合、直ちにコンパイルタスクの実行に切り替える制御を行う。

これにより、本発明のプログラム実行制御装置は、未コンパイルのメソッドを呼び出したタイミングではコンパイル処理を実行せず、コンパイルタスクの実行時にコンパイルするので、プログラムの実行開始当初に頻発するコンパイル処理に伴う立ち上がりの遅延を抑えることができ、実行速度の低下を抑えることができる。

以下、本発明に係るプログラム実行制御装置の各実施形態について説明する。

＜実施形態１＞

＜構成１＞

図２は、実施形態１のプログラム実行制御装置１の機能構成図である。

プログラム実行装置１は、同図に示すように、クラスファイル／クラスライブラリ記憶部２、命令実行処理部３、コンパイル要求管理部４、ネイティブコード記憶部５、コンパイル処理部６、コンパイル済メソッド情報テーブル７及びマルチタスク制御部１９を備える。なお、同図には本発明の特徴となる機能部のみを示しており、図示していない、マル

チタスクＯＳ及びＪＶＭが実装されているコンピュータ装置が通常有する各機能については説明を省略する。

クラスファイル／クラスライブラリ記憶部２は記憶媒体であり、クラスファイル及びクラスライブラリが格納されている。

クラスファイルは、アンテナ２１又はネットワーク２０より取得することができる。

命令実行処理部３は、クラスファイル／クラスライブラリ記憶部２から読み出したバイトコードを逐次解釈してプロセッサに実行させ、メソッド呼び出し時に当該メソッドがコンパイル済みであれば、当該メソッドのネイティブコードをプロセッサに実行させる処理を行う。

ここで、命令実行処理部３に含まれる各機能について説明する。命令実行処理部３には、クラスロード部８、命令取出部９、命令解読部１０、命令実行部１１、メソッド種別判定部１２、及びネイティブコード取出部１３が含まれる。

クラスロード部８は、クラスファイル／クラスライブラリ記憶部２に格納されているクラスファイル又はクラスライブラリをメモリにロードする機能を有する。

命令取出部９は、クラスロード部８によってメモリにロードされたクラスファイル又はクラスライブラリを構成するバイトコードを命令単位で取り出す機能を有する。

命令解読部１０は、命令取出部９によって取り出された命令単位のバイトコードを解読してネイティブコードを選択する機能を有する。

命令実行部１１は、ネイティブコードをプロセッサに実行させる機能を有する。

命令実行部１１により実行される命令が、メソッドを呼び出す呼出コードである場合、メソッド種別判定部１２は、コンパイル済メソッド情報テーブル７を参照して、当該呼出コードによって呼び出されるべきメソッドがコンパイル済みか否かを判定する機能を有する。呼出コードは、Ｊａｓｍｉｎのアセンブラコード表記で言えば、ｉｎｖｏｋｅｖｉｒｔ

ｕａｌ、ｉｎｖｏｋｅｓｐｅｃｉａｌ、ｉｎｖｏｋｓｔａｔｉｃ、ｉｎｖｏｋｅｉｎｔｅｒｆａｃｅといったインストラクションのことである。

呼び出されるべきメソッドがコンパイル済みと判定された場合、ネイティブコード取出部１３は、該当のメソッドのネイティブコードをネイティブコード記憶部５から取り出して命令実行部１１に送る。命令実行部１１は、取り出されたネイティブコードをプロセッサに実行させる。

呼び出されるべきメソッドが未コンパイルと判定された場合、メソッド種別判定部１２は、当該メソッドをコンパイルする要求をコンパイル要求管理部４に通知する。そして、命令取出部９は該当メソッドのバイトコード列を命令単位で逐次取り出し、命令解読部１０は取り出されたバイトコード列を命令単位で逐次解釈し、命令実行部１１は逐次解釈された結果の命令単位のネイティブコードをプロセッサに実行させる。

コンパイル要求管理部４は、メソッド種別判定部１２からコンパイル要求の通知を受けて、受け付けたコンパイル要求に係るメソッドをコンパイルする順番について管理する機能を有する。コンパイル要求管理部４は、登録要求受付部１４、キュー制御部１５及びキュー記憶部１６を有する。

登録要求受付部１４は、メソッド種別判定部１２からコンパイル要求の通知を受け付ける機能を有する。この時に、メソッドに関する情報、例えば、メソッドを特定する識別情報や、メソッドを構成するバイトコード列の格納先アドレス等を取得する。

キュー記憶部１６はＲＡＭ等の記憶媒体である。

キュー制御部１５は、キュー記憶部１６に複数のコンパイル要求情報を記憶させ、これらのコンパイル要求情報をキューで管理する。詳しくは、登録要求受付部１４において受け付けたコンパイル要求をコンパイル要求情報としてキュー記憶部１６に登録し、コンパイル処理部６からコンパイル要求情報の取得要求を受け付けると、キューの先頭位置にあるコンパイル要求情報が示すメソッドをコンパイル処理部６に通知し、コンパイル処理部６においてメソッドのコンパイルが終了するとコンパ

イル処理部６から通知を受けて、当該メソッドのコンパイル要求情報をキュー記憶部１６から削除する。

キュー制御部１５は、コンパイル処理部６からコンパイル要求情報の取得要求を受けて、キューの先頭位置にあるコンパイル要求情報が示すメソッドをコンパイル処理部６に通知する際、コンパイル要求情報に含まれるコンパイル状態を示すビットフラグを立てて、コンパイル中であることを記録する。

キュー制御部１５は、コンパイル要求情報の登録と、登録されているコンパイル要求情報の削除とを排他制御で行う。即ち、同時にコンパイル要求情報の登録と削除を行わないようにしている。また、登録しようとしているコンパイル要求情報がキュー記憶部１６に既に登録されていれば、登録は行わない。

コンパイル処理部６は、コンパイル要求取得部１７及びメソッドコンパイル部１８から構成されており、コンパイル要求取得部１７は、キュー制御部１５にコンパイルすべきメソッドの情報を要求して取得する機能を有し、メソッドコンパイル部１８は、当該メソッドのバイトコード列をネイティブコードに変換する機能を有する。

メソッドコンパイル部１８において生成されたネイティブコードは、ネイティブコード記憶部５に格納され、メソッド名とネイティブコードの格納先アドレスとを示すコンパイル済メソッド情報が、コンパイル済メソッド情報テーブル７に追加される。また、コンパイル要求取得部１７は、メソッドのコンパイルが終了すると、当該メソッドのコンパイルが終了した旨をキュー制御部１５に通知する。

マルチタスク制御部１９は、マルチタスクＯＳの機能であり、優先度ベースのタスクスケジューリングに基づいて、命令実行処理部３により実行される命令実行タスクとコンパイル処理部６により実行されるコンパイルタスクそれぞれにプロセッサ資源の割り当てを行う。

＜データ１＞

ここで各種データについて説明する。

まず、プログラム実行制御装置１により実行されるプログラムについて説明する。

図３は、Ｊａｖａ言語で記述されたソースプログラムの一例を示す図である。なお、本発明の説明上必要がないためｍ１メソッド、ｍ２メソッドの内容については記載を省略している。同図に示すソースプログラムを変換して生成されたバイトコードから成るクラスファイルＡが、プログラム実行制御装置１のクラスファイル／クラスライブラリ記憶部２に記憶される。

クラスファイルをプログラム実行制御装置１が実行する場合、その実行過程においてクラスファイル内の呼出コードによってメソッドが呼び出される。例えば、図３のに示すプログラムの場合だと、まずｉｎｉｔメソッドが１回呼び出され、続いて、ｍ１メソッドが１０回呼び出され、その後、ｍ２メソッドが５回呼び出される。

次に、コンパイル要求情報について説明する。

図４は、キュー記憶部１６に登録されているコンパイル要求情報の具体例を示す図である。

コンパイル要求情報は、次のコンパイル要求情報の位置を示すポインタと、メソッド情報と、コンパイル状態を示すビットフラグから成り、各コンパイル要求情報はポインタによって繋がっており、キューを構成している。

メソッド情報には、コンパイルするメソッドを識別する情報や、メソッドを構成するバイトコード列の格納アドレス等が含まれる。

コンパイル状態を示すビットフラグは、フラグが立っていればコンパイル中、立っていなければ未コンパイルを表す。同図には、わかりやすいように「コンパイル中」、「未コンパイル」と表現している。

同図では、ｉｎｉｔメソッドのコンパイル要求情報がキューの先頭となっており、続いて、ｍ１メソッドのコンパイル要求情報、ｍ２メソッドのコンパイル要求情報の順番で配列されている。よって、この場合、コンパイル処理部６からコンパイルすべきメソッドの情報を要求する通

知がキュー制御部１５に対してなされると、ｉｎｉｔメソッドの情報がコンパイル処理部６に対して通知される。コンパイル処理部６において、ｉｎｉｔメソッドのコンパイルが行われ、コンパイルが終了すると、ｉｎｉｔメソッドのコンパイルが終了した旨の通知をキュー制御部１５に対して行う。

キュー制御部１５は、ｉｎｉｔメソッドのコンパイルが終了した旨の通知を受けると、ｉｎｉｔメソッドのコンパイル要求情報をキュー記憶部１６から削除し、キューの先頭のコンパイル要求情報を示すポインタは、ｍ１メソッドのコンパイル要求情報を示すように更新される。

次にコンパイル済メソッド情報について説明する。

図５は、コンパイル済メソッド情報テーブル７の一例を示す図である。コンパイル済メソッド情報は、コンパイル済みのメソッドを特定する情報（例えば、メソッド名）と、ネイティブコードの格納先アドレスから成る。

＜動作１＞

ここでプログラム実行制御装置１の動作について説明する。

図６は、命令実行処理部３により実行される命令実行タスクのフローを示す図である。

まず、命令実行部１１は、取り出された命令（ネイティブコード）をプロセッサに実行させる（ステップＳ１）。

実行する命令がプログラム終了命令であれば(ステップＳ２：ＹＥＳ)、動作を終了する。実行する命令がプログラム終了命令でなければ（ステップＳ２：ＮＯ）ステップＳ３に進む。

ステップＳ３において、実行する命令が呼出コードであれば（ステップＳ３：ＹＥＳ)、ステップＳ４に進む。ステップＳ３において、実行する命令が呼出コードでなければ（ステップＳ３：ＮＯ)、ステップＳ１に戻る。

ステップＳ４では、メソッド種別判定部１２が、コンパイル済メソッド情報テーブルを参照して呼出対象のメソッドがコンパイル済みか否か

を判定する。呼出対象のメソッドがコンパイル済であれば（ステップＳ４：ＹＥＳ）、ステップＳ５に進む。呼出対象のメソッドがコンパイル済でなければ（ステップＳ４：ＮＯ）、ステップＳ６に進む。

ステップＳ５では、ネイティブコード取出部１３が該当メソッドのネイティブコードをネイティブコード記憶部５から取り出し、取り出したネイティブコードを命令実行部１１に送る。その後、ステップＳ１に戻る。

ステップＳ６では、メソッド種別判定部１２は、コンパイル要求をコンパイル要求管理部４に通知する。続いて、ステップＳ７では、命令取出部９が、該当メソッドのバイトコードを取り出し、命令解読部１０が、取り出したバイトコードの解読を行う。その後、ステップＳ１に戻る。

一方、コンパイル処理部６により実行されるコンパイルタスクは、マルチタスク制御部１９によりプロセッサ資源が割り当てられると、キュー制御部１５に要求し取得したメソッドのバイトコード列のコンパイルを行う。

続いて動作の具体的な例を図２、図４、図５を用いて説明する。図２に示す命令実行部１１が、メソッドｍ２を呼び出す呼出コードをプロセッサに実行させる場合、メソッド種別判定部１２は、当該メソッドｍ２がコンパイル済か否かを図５に示すコンパイル済メソッド情報テーブル７を参照して判定する。

コンパイル済メソッド情報テーブル７には、メソッドｍ２を特定する情報がコンパイル済みメソッドとしてネイティブコードの格納先アドレスと共に記載されているので、メソッド種別判定部１２は、当該メソッドｍ２がコンパイル済であると判定し、ネイティブコード取出部１３に該当のネイティブコードを取り出させる。取り出されたネイティブコードは命令実行部１１に送られて、実行される。

一方、図示されていないメソッドｍ３を呼び出す呼出コードを命令実行部１１がプロセッサに実行させる場合、メソッド種別判定部１２は、図５に示すコンパイル済メソッド情報テーブル７にメソッドｍ３が記載

されていないので未コンパイルと判定し、当該メソッドのコンパイル要求をコンパイル要求管理部４に通知する。そして、当該メソッドのバイトコードを命令単位で逐次取り出して解釈実行する、いわゆるインタプリタ処理を行う。

コンパイル要求管理部４は、図４に示すコンパイル要求情報のキューの最後列、すなわち、メソッドｍ２のコンパイル要求情報の後に、メソッドｍ３のコンパイル要求情報を登録する。

＜実施形態２＞

実施形態２のプログラム実行制御装置は、登録されているコンパイル要求情報のメソッドと同じメソッドについてコンパイル要求がなされる場合、コンパイル要求を受けた回数を要求回数としてコンパイル要求情報毎にカウントして記録し、登録された順番で配列されている各コンパイル要求情報の配列順を、要求回数が多いもの順に入替える順番入替処理を行う。

これにより、頻繁に呼び出されるメソッドを先にコンパイルさせることができ、プログラム実行速度の向上に繋がる。

以下、実施形態２のプログラム実行制御装置について、実施形態１で説明したプログラム実行制御装置１とは異なる部分のみを、構成、データ、動作の各項に分けて説明する。

＜構成２＞

図７は、実施形態２のプログラム実行制御装置１Ａの機能構成図である。

実施形態１のプログラム実行制御装置１と異なる点は、キュー制御部１５Ａ内に加算部２２を備えている点である。

加算部２２は、既に登録されているコンパイル要求情報のメソッドと同一のメソッドのコンパイル要求を受けた場合に、当該コンパイル要求情報のメソッドが未コンパイル状態であれば、当該コンパイル要求情報に含めて記録されている要求回数に１加算した値を算出する機能を有する。キュー制御部１５Ａは、算出した値を要求回数として更新記録する。

＜データ２＞

次に、実施形態２のキュー記憶部１６Ａに登録されている具体的なコンパイル要求情報について説明する。

図８は、キュー記憶部１６Ａに登録されているコンパイル要求情報の具体例を示す図である。実施形態１で説明したコンパイル要求情報と異なる点は、各コンパイル要求情報が２つのポインタを有し、配列位置の前後関係を２つのポインタで示すようにしている点と、要求回数を記録する領域を設けている点である。

同図は、順番入替処理によって要求回数が多いもの順に入替えが行われた後の配列状態を示している。

＜動作２＞

次に、キューの順番入替処理の動作について説明する。

図９は、キュー制御部１５Ａが行うコンパイル要求情報の順番入替処理のフローを示す図である。

まず、登録要求受付部１４でコンパイル要求が受け付けられると、キュー制御部１５Ａは、キューの先頭のコンパイル要求情報を参照する（ステップＳ１１）。

続いて、受け付けられたコンパイル要求に係るメソッドと、参照したコンパイル要求情報に係るメソッドとが同一であれば（ステップＳ１２：ＹＥＳ）、ステップＳ１３に進む。同一でなければ（ステップＳ１２：ＮＯ）、ステップＳ１７に進む。

ステップＳ１７において、全てのコンパイル要求情報を参照し終わっていれば（ステップＳ１７：ＹＥＳ）、ステップＳ１９に進み、まだ、全てのコンパイル要求情報を参照していなければ（ステップＳ１７：ＮＯ）、ステップＳ１８に進む。

ステップＳ１８において、参照したコンパイル要求情報のポインタが示す、１つ後の配列位置のコンパイル要求情報を参照し、ステップＳ１２に進む。

ステップＳ１９において、受け付けたコンパイル要求に係るメソッド

のコンパイル要求情報を、キューの最後列に登録する。

ステップＳ１３において、参照したコンパイル要求情報のメソッドが未コンパイルであるかコンパイル中かをビットフラグで確認する。コンパイル中であれば（ステップＳ１３：ＮＯ）、入替処理を終了し、未コンパイルであれば（ステップＳ１３：ＹＥＳ）、ステップＳ１４に進む。

ステップＳ１４において、参照したコンパイル要求情報の要求回数に１加算した値を要求回数として更新する。続いて、ステップＳ１５において、１つ前の配列位置にあるコンパイル要求情報の要求回数と、更新した要求回数の大きさを比較する。

その結果、１つ前の配列位置にあるコンパイル要求情報の要求回数より、更新した要求回数の方が大きな値であれば（ステップＳ１５：ＹＥＳ）、ステップＳ１６に進む。そうでなければ（ステップＳ１５：ＮＯ）、順番入替処理を終了する。

ステップＳ１６において、１つ前の配列位置にあるコンパイル要求情報と、要求回数を更新したコンパイル要求情報の配列順番を入替える。すなわち、入替える２つのコンパイル要求情報及びその前後に位置する各コンパイル要求情報のポインタ情報の変更を行う。その後、順番入替処理を終了する。

＜実施形態３＞

実施形態３のプログラム実行制御装置は、各メソッドの優先度が記載されている優先度情報テーブルをクラスファイルから取得し、取得した優先度情報テーブルを参照して、コンパイルするメソッドの優先度を特定し、優先度に基づいてコンパイル要求情報の配列位置を決定する処理を行う。

これにより優先度の高いメソッドからコンパイルされるので、プログラム実行速度の向上に繋がる。

以下、実施形態３のプログラム実行制御装置について、実施形態１で説明したプログラム実行制御装置１とは異なる部分のみを、構成、データ、動作の各項に分けて説明する。

＜構成３＞

図１０は、実施形態３のプログラム実行制御装置１Ｂの機能構成図である。

実施形態１のプログラム実行制御装置１と異なる点は、優先度情報テーブル２３を取得する点と、キュー制御部１５Ｂが、メソッドの優先度に基づいて登録するコンパイル要求情報の、キューにおける登録位置を決定している点である。

＜データ３＞

次に各種データについて説明する。

図１１は、クラスファイルから取得した優先度情報テーブル２３の具体的な内容を示す図である。優先度情報テーブル２３では、メソッドと優先度とが対応付けられている。ここでいう優先度とは、コンパイル処理を優先的に行うべき度合いを示す指標であり、値が大きい方が優先度の高いものとしている。よって、優先度の高いものから並べると、メソッドｍ３、メソッドｉｎｉｔ、メソッドｍ２、メソッドｍ１の順番になる。なお、本実施形態では、各メソッドの優先度はそれぞれ異なる値となっている。

図１２は、キュー記憶部１６Ｂに登録されているコンパイル要求情報の具体例を示す図である。同図に示すように、優先度が高いコンパイル要求情報が先になるように配列されている。

＜動作３＞

次にキュー制御部１５Ｂによるコンパイル要求情報の登録位置決定処理について説明する。

図１３は、キュー制御部１５Ｂが行うコンパイル要求情報の登録位置決定処理のフローを示す図である。

まず、登録要求受付部１４Ｂにおいてコンパイル要求が受け付けられると、キュー制御部１５Ｂは、キューの先頭のコンパイル要求情報を参照する（ステップＳ２１）。

続いて、受け付けられたコンパイル要求に係るメソッドと、参照した

コンパイル要求情報に係るメソッドとが同一であれば（ステップＳ２２：ＹＥＳ）、登録位置決定処理を終了する。同一でなければ（ステップＳ２２：ＮＯ）、ステップＳ２３に進む。

ステップＳ２３において、登録するメソッドの優先度と、参照したコンパイル要求情報に記載されている優先度とを比較し、登録するメソッドの優先度の方が高ければ（ステップＳ２３：ＹＥＳ）、ステップＳ２４に進む。参照したコンパイル要求情報に記載されている優先度の方が高ければ、（ステップＳ２３：ＮＯ）、ステップＳ２５に進む。

ステップＳ２５において、全てのコンパイル要求情報を参照し終わっていれば（ステップＳ２５：ＹＥＳ）、ステップＳ２６に進み、まだ、全てのコンパイル要求情報を参照していなければ（ステップＳ２５：ＮＯ）、ステップＳ２７に進む。

ステップＳ２７において、参照したコンパイル要求情報のポインタが示す、１つ後の配列位置のコンパイル要求情報を参照し、ステップＳ２２に進む。

ステップＳ２６において、受け付けたコンパイル要求に係るメソッドのコンパイル要求情報を、キューの最後列に登録して、登録位置決定処理を終了する。

ステップＳ２４において、登録するメソッドのコンパイル要求情報を、参照したコンパイル要求情報の１つ前の配列位置に登録して、登録位置決定処理を終了する。

例えば、図１２に示すコンパイル要求情報のキューに、優先度２のメソッドｍ３のコンパイル要求情報を登録する場合、登録位置決定処理により、メソッドｍ２のコンパイル要求情報とメソッドｍ１のコンパイル要求情報との間にメソッドｍ３のコンパイル要求情報が登録されることになるが、具体的には、メソッドｍ２のコンパイル要求情報及びメソッドｍ１のコンパイル要求情報のポインタそれぞれが、メソッドｍ３のコンパイル要求情報の登録位置を示すように更新することで、メソッドｍ３のコンパイル要求情報の登録位置が決定される。

＜実施形態４＞

実施形態４のプログラム実行制御装置は、優先度が高、中、低と付けられている３つのキューを設けて、優先度の高いキューに登録されているコンパイル要求情報のメソッドから順番にコンパイルする。すなわち、優先度高のキューの先頭位置に登録されているコンパイル要求情報のメソッドから順番にコンパイルをし、優先度高のキューに登録されているコンパイル要求情報のメソッドが全てコンパイルされると、次に優先度中のキューの先頭に登録されているコンパイル要求情報のコンパイルを行い、優先度中のキューに登録されているコンパイル要求情報のメソッドが全てコンパイルされると、次に優先度低のキューの先頭に登録されているコンパイル要求情報のコンパイルを行う。

優先度高のキューには、プログラム実行制御装置のＯＳ起動と共にキュー記憶部に一括登録されるコンパイル要求情報が登録され、優先度中のキューには、優先度情報テーブルに優先度が示されているメソッドのコンパイル要求情報が登録され、優先度低のキューには、優先度高のキュー及び優先度中のキューに登録されるメソッド以外のメソッドが登録される。

以下、実施形態４のプログラム実行制御装置について、実施形態１から実施形態３までで説明したプログラム実行制御装置とは異なる部分のみを、構成、データの各項に分けて説明する。

＜構成４＞

図１４は、実施形態４のプログラム実行制御装置１Ｃの機能構成図である。

実施形態１のプログラム実行制御装置１と異なる点は、キュー制御部１５Ｃが優先度の異なる３つのキューを制御している点と、一括登録部２５を新たに備えて、優先的にコンパイルすべきメソッドのメソッド名や格納先等を示すメソッド情報のテーブルである一括登録情報２４に基づいて、ＯＳ起動と共にキュー記憶部１６Ｃの優先度高のキューに、これらのメソッドのコンパイル要求情報を登録する点と、実施形態２で説

明した順番入替処理や、実施形態３で説明した登録位置決定処理を行ってコンパイル要求情報を登録している点である。

＜データ４＞

次に各種データについて説明する。

図１５は、一括登録情報２４の具体的な内容を示す図である。一括登録情報２４の各メソッドは、ユーザによりコンパイルすべきメソッドとして指定されたメソッドや、プログラム実行シミュレーションの結果、頻繁に呼び出されることがわかっているメソッド等を想定している。

図１６は、キュー記憶部１６Ｃに登録されているコンパイル要求情報の具体例を示す図である。同図に示すように、優先度高の第一キュー、優先度中の第二キュー、優先度低の第三キューが設けられており、第一キューには、図１５で示した一括登録情報２４に記載されていたメソッドｍ４、メソッドｍ５、メソッドｍ６それぞれのコンパイル要求情報が登録されている。

また、第二キューには、優先度が付けられている各種メソッドが優先度順で登録されている。そして、第三キューには、第一キュー及び第二キューに登録されることのないメソッドのコンパイル要求情報が順番入替処理により動的に順番を変えながら配列されている。

＜実施形態５＞

実施形態５のプログラム実行制御装置は、関連メソッドテーブルをクラスファイルから取得し、コンパイル処理部６においてコンパイルを行うメソッドと関連付けされているメソッドがないかどうかを関連メソッドテーブルから検索し、関連付けされているメソッドが検出されれば、その検出されたメソッドの登録要求を行う。

以下、実施形態５のプログラム実行制御装置について、実施形態１で説明したプログラム実行制御装置１とは異なる部分のみを、構成、データ、動作の各項に分けて説明する。

＜構成５＞

図１７は、実施形態５のプログラム実行制御装置１Ｄの機能構成図で

ある。

実施形態１のプログラム実行制御装置１と異なる点は、関連メソッド検索処理部２６を備えている点である。

関連メソッド検索処理部２６は、関連メソッドテーブル２８を取得し、コンパイル要求取得部１７から通知されたメソッドと関連付けされているメソッドがないかどうかを、取得した関連メソッドテーブル２８から検索する検索部２７とを有する。関連するメソッドが検出されれば、検出されたメソッドをコンパイル要求されたものとして、登録要求受付部１４Ｄに通知する。

＜データ５＞

図１８は、関連メソッドテーブル２８の具体例を示す図である。

メソッドinitと関連付けられているメソッドは、メソッドｍ１、メソッドｍ２であり、メソッドｍ３と関連付けられているメソッドは、メソッドｍ４、メソッドｍ５である。

＜動作５＞

次に、関連メソッドの検索から登録までの処理である関連メソッド検索処理の動作について説明する。

図１９は、関連メソッド検索処理部２６においてなされる関連メソッド検索処理のフローを示す図である。

まず、コンパイル要求取得部１７からメソッドが通知されると、検索部２７は、関連メソッドテーブル２８を参照して、関連メソッドがあるかないかを検索する（ステップＳ３１）。

関連メソッドが検出されれば（ステップＳ３２：ＹＥＳ）、ステップＳ３３に進む。検出されなければ（ステップＳ３２：ＮＯ）、関連メソッド検索処理を終了する。

ステップＳ３３において、検出された関連メソッドについて、コンパイル要求がなされたものとして、登録要求受付部１４Ｄに通知し、関連メソッド検索処理を終了する。

ここで、具体的な動作を図１８を用いて説明する。関連メソッド検索

処理部２６は、コンパイル要求取得部１７からメソッドｉｎｉｔが通知されると、関連メソッドテーブル２８を参照して、関連メソッドがあるかないかを検索する。その結果、図１８に示すように、メソッドｉｎｉｔには、メソッドｍ１、メソッドｍ２が関連付けられているため、メソッドｍ１、メソッドｍ２を登録要求受付部１４Ｄに通知する。

＜実施形態６＞

実施形態６のプログラム実行制御装置は、実施形態３で説明したように、各メソッドの優先度に基づいて登録位置決定処理を行うが、更に、コンパイルするメソッドの優先度と命令実行タスクの優先度との比較を行い、その結果、コンパイルするメソッドの優先度の方が命令実行タスクの優先度より高ければ、当該メソッドの優先度をコンパイルタスクの優先度とする優先度継承処理を行う。

以下、実施形態６のプログラム実行制御装置について、実施形態３で説明したプログラム実行制御装置１Ｂとは異なる部分のみを、構成、動作の各項に分けて説明する。

＜構成６＞

図２０は、実施形態６のプログラム実行制御装置１Ｅの機能構成図である。

実施形態３のプログラム実行制御装置１Ｂと異なる点は、優先度継承部２９を備えている点である。

優先度継承部２９は、機能的には優先度比較部３０とタスク優先度変更部３１を含む。

優先度比較部３０は、コンパイル要求取得部１７からコンパイル開始するメソッドの優先度の通知を受けて、当該メソッドの優先度と命令実行タスクの優先度とを比較する。

比較した結果、コンパイル開始するメソッドの優先度の方が高い場合、タスク優先度変更部３１は、マルチタスク制御部１９Ｅに当該メソッドの優先度を、コンパイルタスクの優先度とする指示を行う。

マルチタスク制御部１９Ｅは、タスク優先度変更部３１からの指示に

基づいて、コンパイルタスクの優先度を変更する。

＜動作６＞

次に優先度継承処理について説明する。

図２１は、優先度継承処理の動作を説明するためのフロー図である。

まず、優先度継承部２９は、コンパイル要求取得部１７からコンパイル開始するメソッドの優先度の通知を受けて、優先度比較部３０において当該メソッドの優先度と命令実行タスクの優先度とを比較する（ステップＳ４１）。

その結果、メソッドの優先度の方が命令実行タスクの優先度より高ければ（ステップＳ４２：ＹＥＳ）、タスク優先度変更部３１は、コンパイルタスクの優先度をメソッドの優先度に変更する指示をマルチタスク制御部１９Ｅに通知する（ステップＳ４３）。命令実行タスクの優先度の方がメソッドの優先度より高ければ（ステップＳ４２：ＮＯ）、ステップＳ４１に戻る。

例えば、命令実行タスクの優先度のデフォルト値が６で、コンパイルタスクの優先度のデフォルト値が２であり、コンパイルするメソッドの優先度が７である場合、優先度比較部３０においてコンパイルするメソッドの優先度と命令実行タスクの優先度を比較すると、７＞６より、コンパイルするメソッドの優先度の方が高いので、タスク優先度変更部３１は、コンパイルタスクの優先度をメソッドの優先度に変更する指示をマルチタスク制御部１９Ｅに対して行う。マルチタスク制御部１９Ｅは、指示に基づいてコンパイルタスクの優先度を７とし、これは、命令実行タスクのデフォルト優先度６より高い値なので、コンパイルタスクに対してプロセッサ資源を優先的に割り当てる。

＜補足＞

本発明は上述の各実施形態に限られないことは勿論である。即ち、

（１）実施形態１の説明では、メソッド種別判定部１２において、呼び出すメソッドがコンパイル済でなければ、直ちに当該メソッドのインタプリタ処理を行っていたが、呼び出すメソッドがコンパイル中であれば、

そのコンパイルの終了を待って、コンパイルされたメソッドのネイティブコードを実行するようにしてもよい。その場合の動作について図を用いて説明する。

図２２は、呼び出すメソッドがコンパイル中であれば待つ場合の命令実行タスクのフローを示す図である。

まず、命令実行部１１は、取り出された命令をプロセッサに実行させる（ステップＳ５１）。実行する命令がプログラム終了命令であれば（ステップＳ５２：ＹＥＳ）、動作を終了する。実行する命令がプログラム終了命令でなければ（ステップＳ５２：ＮＯ）ステップＳ５３に進む。

ステップＳ５３において、実行する命令が呼出コードであれば（ステップＳ５３：ＹＥＳ）、ステップＳ５４に進む。ステップＳ５３において、実行する命令が呼出コードでなければ（ステップＳ５３：ＮＯ）、ステップＳ５１に戻る。

ステップＳ５４では、メソッド種別判定部１２が、コンパイル済メソッド情報テーブルを参照して呼出対象のメソッドがコンパイル済みか否かを判定する。呼出対象のメソッドがコンパイル済であれば（ステップＳ５４：ＹＥＳ）、ステップＳ５５に進む。呼出対象のメソッドがコンパイル済でなければ（ステップＳ５４：ＮＯ）、ステップＳ５６に進む。

ステップＳ５５では、ネイティブコード取出部１３が該当メソッドのネイティブコードをネイティブコード記憶部５から取り出し、取り出したネイティブコードを命令実行部１１に送る。その後、ステップＳ５１に戻る。

ステップＳ５６では、メソッド種別判定部１２は、キュー制御部１５にキューを参照させて、呼出対象のメソッドが現在コンパイル中か否かを確認させる。その結果、呼出対象のメソッドが現在コンパイル中であれば（ステップＳ５６：ＹＥＳ）、コンパイルが終了するまで待ち（ステップＳ５７）、そうでなければ（ステップＳ５６：ＮＯ）、コンパイル要求をコンパイル要求管理部４に通知する。続いて、ステップＳ５９では、命令取出部９が、該当メソッドのバイトコードを取り出し、命令解読部

１０が、取り出したバイトコードの解読を行う。その後、ステップＳ５１に戻る。

（２）図１では、１つの命令実行タスクと１つのコンパイルタスクの実行状態推移を示したが、命令実行タスク及びコンパイルタスクは複数であってもよく、この場合、各タスクにそれぞれ異なる優先度が付けられていてもよい。

上記各実施形態では、コンパイルタスクの優先度より命令実行タスクの優先度を高くしていたが、複数のコンパイルタスクを設ける場合、特定のコンパイルタスクを例外として、当該コンパイルタスクの優先度を命令実行タスクの優先度より高く設定していてもよい。

例えば、実施形態４では、優先度が高、中、小と付けられている３つのキューを設けていたが、それぞれ優先度が異なるコンパイルタスクのキューであるとして、優先度高の第一キューに登録されているコンパイル要求情報のメソッドをコンパイルするコンパイルタスクの優先度を、命令実行タスクの優先度より高く設定しておくことが考えられる。

（３）上述した各実施形態のプログラム実行制御装置は、シングルプロセッサ資源の各タスクへの割り当てをスケジューリングして、各タスクを擬似並行処理しているものとして説明したが、本発明のプログラム実行制御装置は、マルチプロセッサ、或いは複数の独立したプロセッサを使用して、各タスクを同時に並行して実行するものであってもよい。

（４）本発明は、ＪｉＴコンパイラ付ＪＶＭが実装されたプログラム実行制御装置の適用のみに限られず、コンパイラ機能を備えた、中間コードを解釈実行する仮想マシンが実装されているプログラム実行制御装置全般に適用可能である。

（５）各実施形態では、メソッドをコンパイルする順番をキューで管理していたが、本発明に係るプログラム実行制御装置は、必ずしもキューでコンパイルする順番を管理する必要はない。例えば、プログラム毎、或いはクラスファイル毎に予め設定されている閾値を取得して、コンパイル要求回数が閾値を超えたものからコンパイルするようにしてもよい。

また、プログラム実行開始から所定時間毎に、その時に登録されている全てのコンパイル要求情報のうち、最も優先度が高いメソッドをコンパイルするようにしてもよいし、コンパイルタスクにプロセッサ資源が割り当てられる都度、その時に登録されているコンパイル要求情報のうち、最も優先度の高いメソッドをコンパイルするようにしてもよい。

（６）実施形態４では、３つの優先度の異なるキューを用いていたが、３つに限られず、２つでも、４つでもよい。

（７）本発明に係るプログラム実行制御装置は、一般的なＪＶＭが備えているガーベジコレクション機能を有しており、ガーベジコレクションは、ガーベジコレクションタスクとして命令実行タスク及びコンパイルタスクと並行して実行されることを想定する。コンパイルしたネイティブコードのうち、再び取り出されることのないもの、使用頻度が小さいものについては、ガーベジコレクションによって破棄されるようにしてもよい。

（８）各実施形態に示したプログラム実行制御装置による処理手順（図６、図９、図１３、図１９、図２１、図２２に示した手順等）を、プログラム実行機能を備える機器等に実行させるための制御プログラムを、記録媒体に記録し又は各種通信路等を介して、流通させ頒布することもできる。このような記録媒体には、ＩＣカード、光ディスク、フレキシブルディスク、ＲＯＭ等がある。流通、頒布された制御プログラムは、ＲＯＭを備える機器等にインストールされることにより利用に供され、その機器等はその制御プログラムの実行により各実施形態で示したようなプログラム実行制御装置の機能を実現する。

産業上の利用可能性

本発明に係るプログラム実装制御装置は、プロセッサを備えた様々な電子機器に実装される仮想マシンとして適用できる。

請 求 の 範 囲

1．バイトコード列を呼び出す呼出コードを含む1又は複数のバイトコード列からなるプログラムをプロセッサに実行させるプログラム実行制御装置であって、

前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定手段と、

呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第1手段と、

呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第2手段と、

前記第1手段によりなされるバイトコード列の逐次解釈実行又は前記第2手段によりなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第3手段とを備える

ことを特徴とするプログラム実行制御装置。

2．前記プログラム実行制御装置は、マルチタスクオペレーティングシステムの制御下で動作し、前記第1手段によりなされる前記バイトコード列の実行又は前記第2手段によりなされる前記ネイティブコードの実行と、前記第3手段によりなされる前記変換処理の実行は、それぞれ異なるタスクとして実行され、前記第1手段及び前記第2手段のタスク実行は、前記第3手段のタスク実行より優先度が高いことを特徴とする請求の範囲第1項記載のプログラム実行制御装置。

3．前記第１手段又は前記第２手段のタスク実行において待ち状態が発生した場合、前記第３手段のタスク実行に切り替える切替手段を更に備える

ことを特徴とする請求の範囲第２項記載のプログラム実行制御装置

4．前記第１手段によりなされる前記要求に応じて、当該要求に係るバイトコード列をネイティブコードに変換するための情報であるコンパイル要求情報を記憶手段に登録し管理する要求管理手段を更に備え、

前記第３手段は、前記第１手段によりなされるバイトコード列の逐次解釈実行又は前記第２手段によりなされる前記ネイティブコードの実行と並行して、前記要求管理手段によって前記記憶手段に登録されているコンパイル要求情報に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる

ことを特徴とする請求の範囲第３項記載のプログラム実行制御装置。

5．前記要求管理手段は、前記要求を先に受けたものからの順番となるキューで各コンパイル要求情報を登録して管理し、

前記第３手段は、前記キューの先頭位置にあるコンパイル要求情報から順番に前記変換処理を前記プロセッサに実行させる

ことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

6．前記要求管理手段は、前記第１手段によりなされる前記要求に係るバイトコード列のコンパイル要求情報が前記記憶手段に既に登録されている場合に、２重登録を行わないことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

7．各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段を更に備え、

前記要求管理手段は、

前記第１手段によりなされる前記要求に応じて、取得された優先度情報を参照し、当該要求に係るバイトコード列の優先度を特定する特定手段と、

前記記憶手段に登録されている各コンパイル要求情報に係るバイトコード列の優先度と、前記特定手段により特定された優先度とを比較する比較手段と、

比較した結果に基づいて、優先度の高いもの順の配列となるように、前記要求に係るバイトコード列のコンパイル要求情報のキュー位置を決定する決定手段とを備える

ことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

８．あるバイトコード列と関連する関連バイトコード列を対応付けている対応情報を取得する対応情報取得手段と、

前記対応情報を参照して、前記要求に係るバイトコード列と対応付けられている関連バイトコード列があるか否かを検索する検索手段とを更に備え、

前記検索手段による検索結果、関連バイトコード列が検出された場合、前記要求管理手段は、検出された関連バイトコード列のコンパイル要求情報を記憶手段に登録する

ことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

９．各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段を更に備え、

前記第３手段は、前記記憶手段に複数登録されているコンパイル要求情報に係るバイトコード列のうち、取得された前記優先度情報に基づいて優先度の高いものから順番に前記変換処理を前記プロセッサに実行させる

ことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

１０．前記要求管理手段は、

繰り返し前記要求がなされるバイトコード列については、その要求回数を計数して、計数した要求回数を、当該バイトコード列のコンパイル要求情報に含めて前記記憶手段に記録する回数記録手段と、

前記要求回数の閾値を取得する取得手段とを備え、

前記第３制御手段は、前記要求回数が閾値を超えたものから順番に前記変換処理を前記プロセッサに実行させる

ことを特徴とする請求の範囲第４項記載のプログラム実行制御装置。

１１．前記要求管理手段は、

繰り返し前記要求がなされるバイトコード列については、その要求回数を計数し、計数した要求回数を当該バイトコード列のコンパイル要求情報に含めて前記記憶手段に記録する回数記録手段と、

記録した各コンパイル要求情報の要求回数を比較して、要求回数が多いもの順となるように各コンパイル要求情報のキュー位置を入替える入替手段とを更に備える

ことを特徴とする請求の範囲第５項記載のプログラム実行制御装置。

１２．前記要求管理手段は、優先度の異なる複数のキューを管理しており、

前記第３手段は、優先度が最も高いキューに登録されているコンパイル要求情報に係るバイトコード列から順番に前記変換処理を前記プロセッサに実行させる

ことを特徴とする請求の範囲第５項記載のプログラム実行制御装置。

１３．前記プログラムの実行前に、ネイティブコードに変換すべき複数のバイトコード列を示す特別要求情報を取得する特別要求情報取得手段を更に備え、

前記要求管理手段は、

取得された特別要求情報に示される各バイトコード列のコンパイル要求情報を、優先度が最も高いキューに一括登録して管理する

ことを特徴とする請求の範囲第１２項記載のプログラム実行制御装置。

１４．前記判定手段により、呼び出されるべきバイトコード列がネィティブコードに未変換であると判定されたときに、当該バイトコード列が現在、前記変換処理の途中であるか否かを判定する第２判定手段と、

前記バイトコード列が現在、前記変換処理の途中であると判定された場合に、当該変換処理が終了するのを待って、変換されたネイティブコードを前記プロセッサに実行させる第４手段とを備える

ことを特徴とする請求の範囲第１項記載のプログラム実行制御装置。

１５．前記第１手段によりなされる前記要求に応じて、当該要求に係るバイトコード列をネイティブコードに変換するための情報であるコンパイル要求情報を記憶手段に登録し管理する要求管理手段を更に備え、

前記第３手段は、前記第１手段によりなされるバイトコード列の逐次解釈実行又は前記第２手段によりなされる前記ネイティブコードの実行と並行して、前記要求管理手段によって前記記憶手段に登録されているコンパイル要求情報に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる

ことを特徴とする請求の範囲第１項記載のプログラム実行制御装置。

１６．各バイトコード列の優先度を示す優先度情報を取得する優先度情報取得手段と、

前記変換処理を行うバイトコード列の優先度と前記命令実行タスクの優先度とを比較する比較手段と、

比較した結果、前記バイトコード列の優先度の方が命令実行タスクの優先度より高い場合に、前記コンパイルタスクの優先度を一時的に高くする優先度変更手段とを更に備える

ことを特徴とする請求の範囲第２項記載のプログラム実行制御装置。

１７．前記第１手段によりなされる前記要求に応じて、当該要求に係るバイトコード列をネイティブコードに変換するための情報であるコンパイル要求情報を記憶手段に登録し管理する要求管理手段を更に備え、

前記第３手段は、前記第１手段によりなされるバイトコード列の逐次解釈実行又は前記第２手段によりなされる前記ネイティブコードの実行と並行して、前記要求管理手段によって前記記憶手段に登録されているコンパイル要求情報に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる

ことを特徴とする請求の範囲第２項記載のプログラム実行制御装置。

１８．バイトコード列を呼び出す呼出コードを含む１又は複数のバイトコード列からなるプログラムをプロセッサに実行させるプログラム実行制御方法であって、

前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定ステップと、

呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、

呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、

前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含む

ことを特徴とするプログラム実行制御方法。

１９．バイトコード列を呼び出す呼出コードを含む１又は複数のバイトコード列からなるプログラムをプロセッサに実行させる制御プログラムであって、

前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定ステップと、

呼び出されるべきバイトコード列がネィティブコードに未変換であると判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、

呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、

前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含む

ことを特徴とする制御プログラム。

２０．バイトコード列を呼び出す呼出コードを含む１又は複数のバイトコード列からなるプログラムをプロセッサに実行させる制御プログラムを記録した記録媒体であって、

前記制御プログラムは、

前記プログラムの実行過程において呼出コードが実行対象とされる度に、呼び出されるべきバイトコード列が前記プロセッサ用のネイティブコードに変換済みか否かを判定する判定ステップと、

呼び出されるべきバイトコード列がネィティブコードに未変換である

と判定された場合に、当該バイトコード列を逐次解釈して前記プロセッサに実行させ、且つ、当該バイトコード列をネイティブコードに変換する要求を行う第１ステップと、

呼び出されるべきバイトコード列がネィティブコードに変換済みであると判定された場合に、当該ネイティブコードをプロセッサに実行させる第２ステップと、

前記第１ステップにおいてなされるバイトコード列の逐次解釈実行又は前記第２ステップにおいてなされる前記ネイティブコードの実行と並行して、前記要求に係るバイトコード列をネイティブコードに変換する変換処理をプロセッサに実行させる第３ステップとを含む

ことを特徴とする記録媒体。

図1

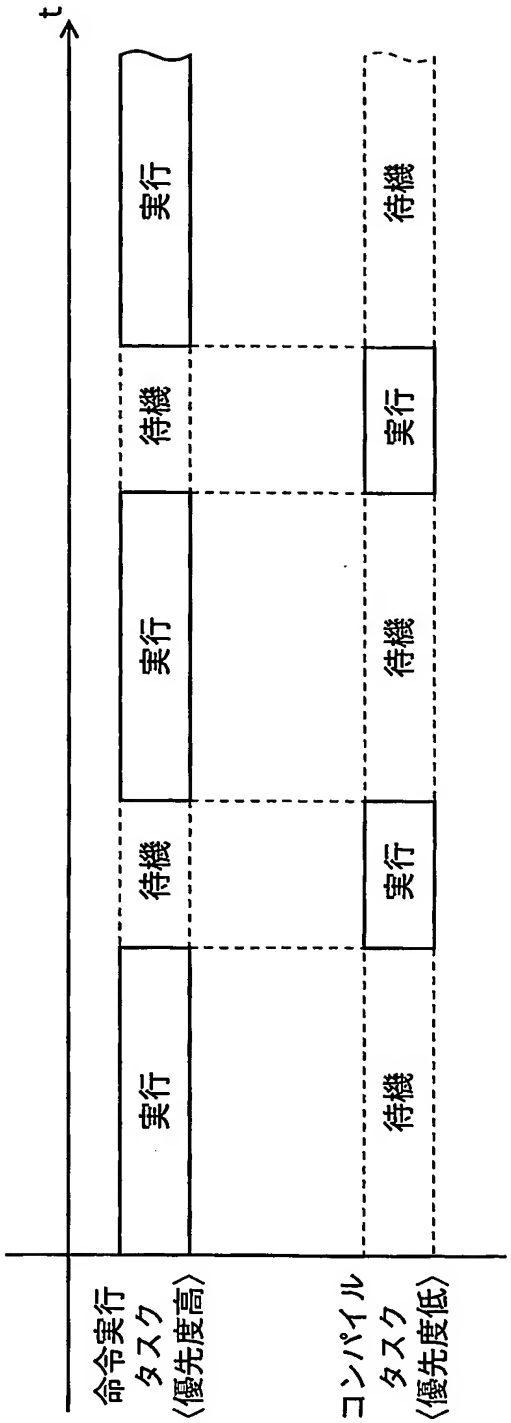


図2

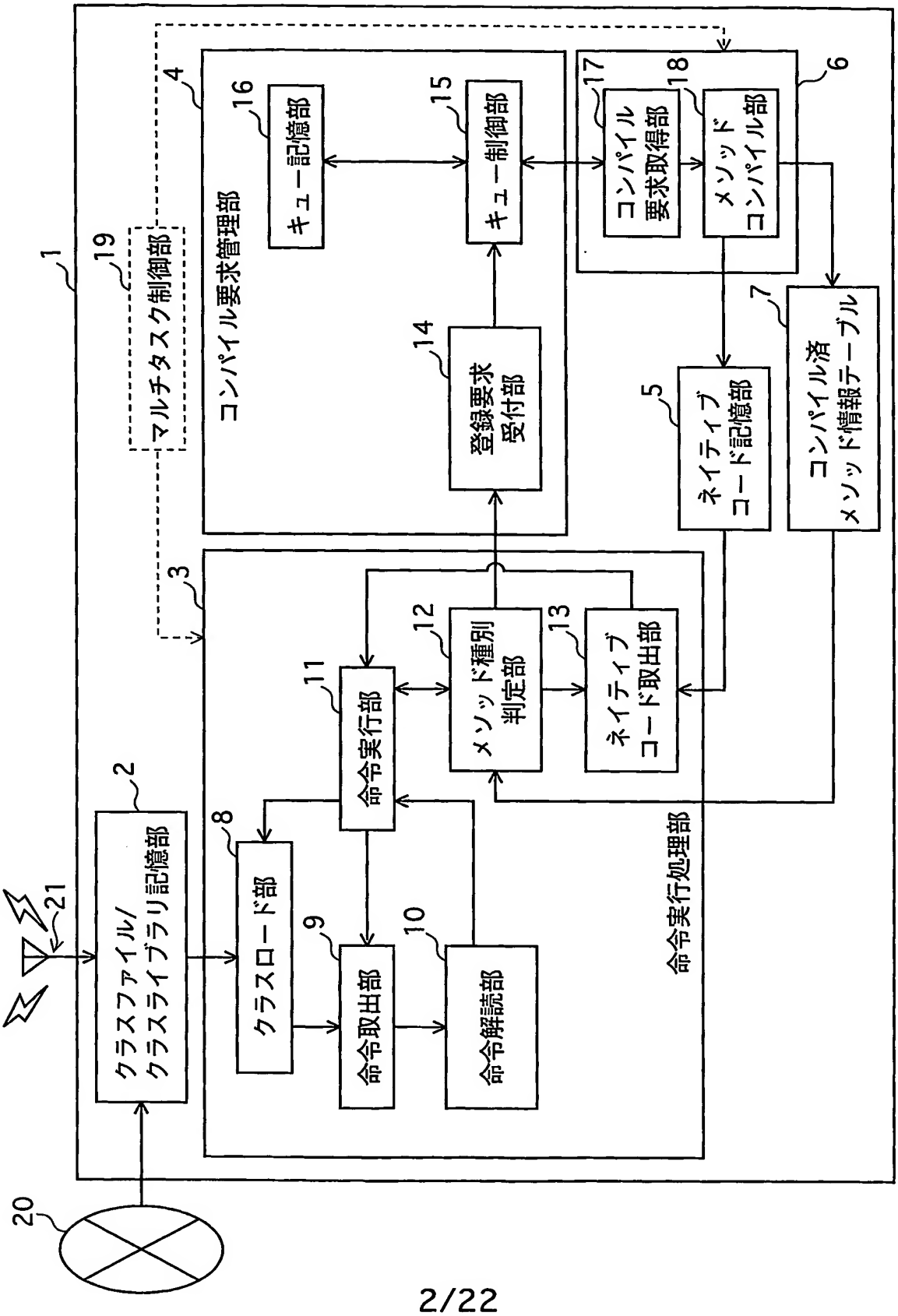

1
2
クラスファイル/
クラスライブラリ記憶部
3
4
コンパイル要求管理部
5
ネイティブ
コード記憶部
6
7
コンパイル済
メソッド情報テーブル
8
クラスロード部
9
命令取出部
10
命令解読部
11
命令実行部
12
メソッド種別
判定部
13
ネイティブ
コード取出部
14
登録要求
受付部
15
キュー制御部
16
キュー記憶部
17
コンパイル
要求取得部
18
メソッド
コンパイル部
19
マルチタスク制御部
20
21
命令実行処理部

図3

```
public class  A
        public void init ()  {
            int i ;
            for (i=0; i<10  ; i ++)  {
                    m1() ;
            }

            for (i=0; i<5  ; i ++)  {
                    m2() ;
            }
        }
        public void m1()  {
                ·····
        }

        public void m2()  {
                ·····
        }
}
```

図4

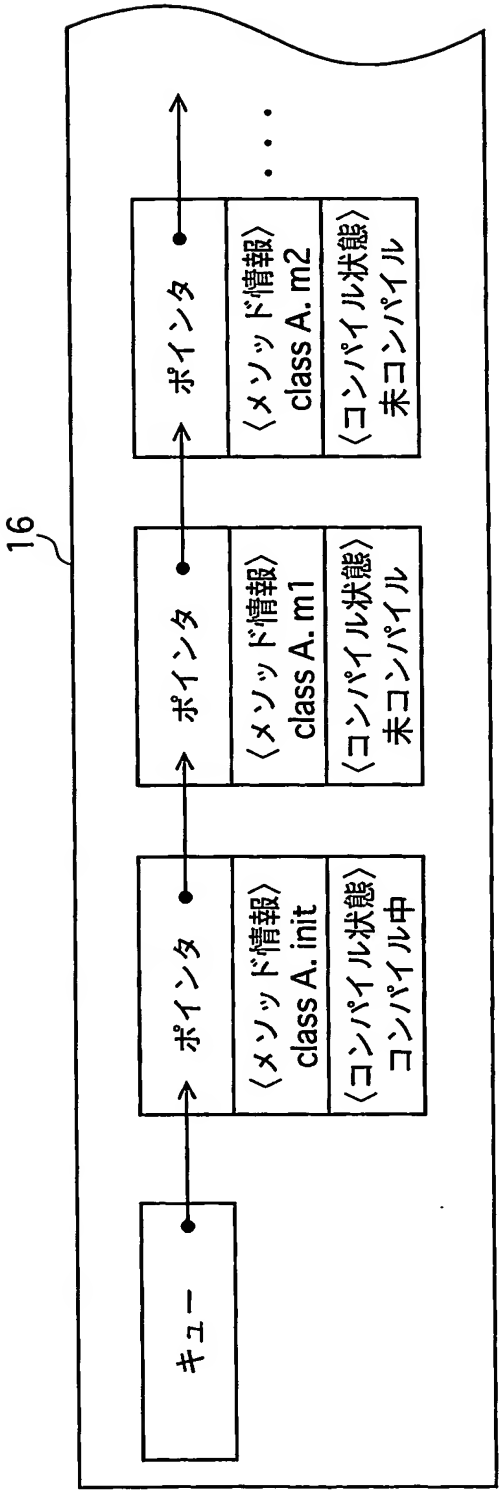

16
キュー
ポインタ
〈メソッド情報〉
class A. init
〈コンパイル状態〉
コンパイル中
ポインタ
〈メソッド情報〉
class A. m1
〈コンパイル状態〉
未コンパイル
ポインタ
〈メソッド情報〉
class A. m2
〈コンパイル状態〉
未コンパイル
・・・

図5

7

コンパイル済 メソッド名	格納先アドレス
init	0x48000
m 1	0x4a000
m 2	0x4c000

図6

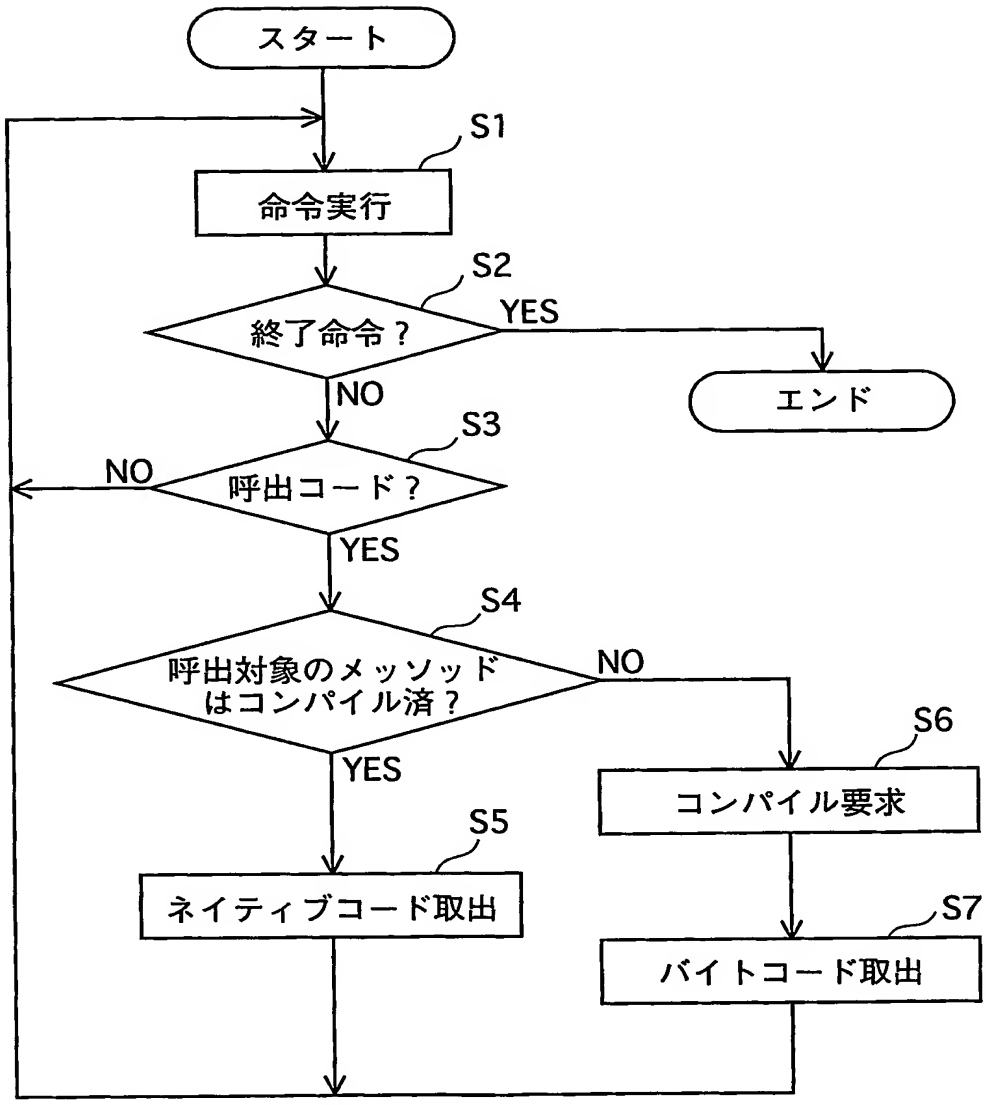

スタート
S1
命令実行
S2
終了命令？
YES
エンド
NO
S3
NO
呼出コード？
YES
S4
呼出対象のメッソッド
はコンパイル済？
NO
YES
S6
コンパイル要求
S5
ネイティブコード取出
S7
バイトコード取出

図7

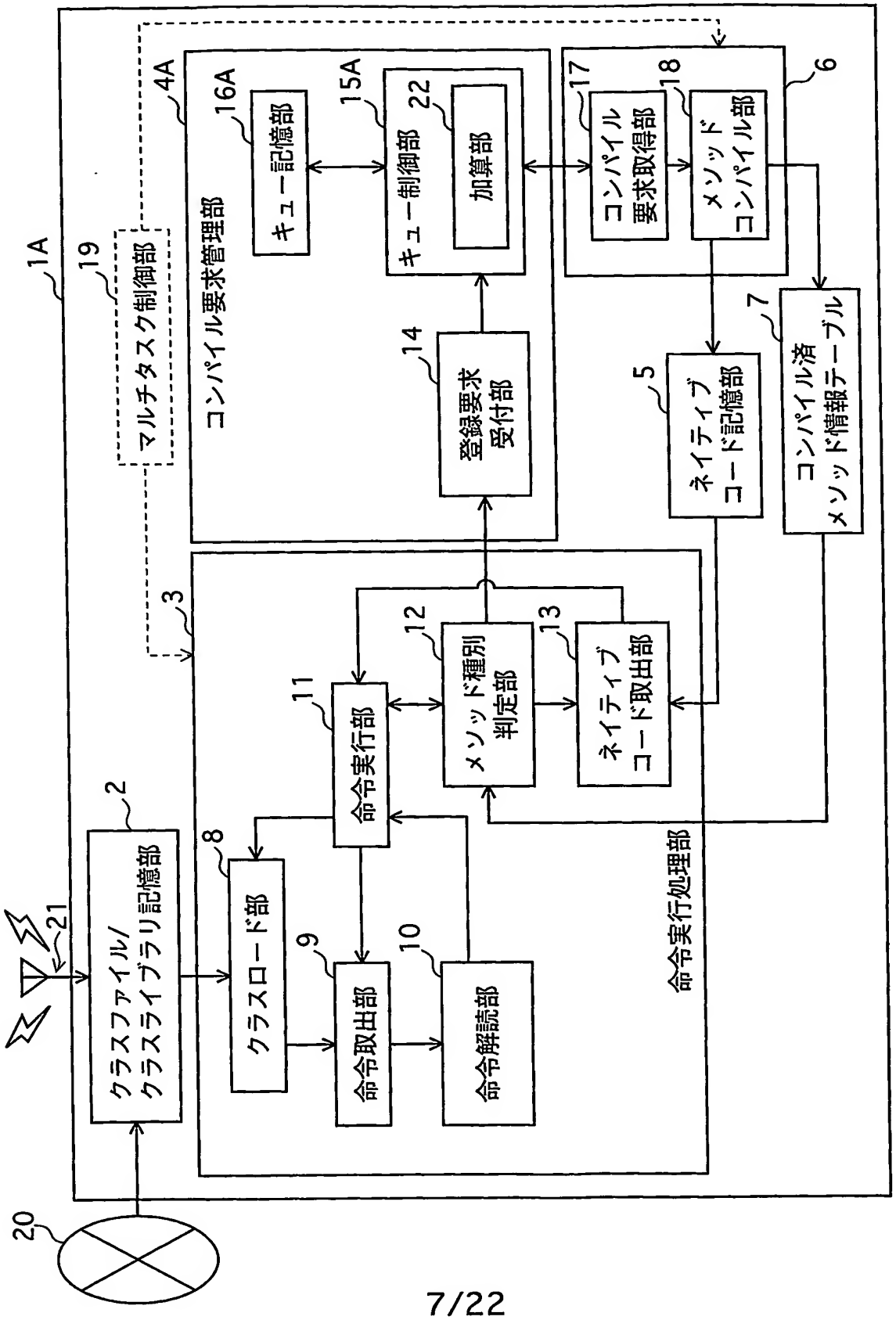

1A
19
マルチタスク制御部
4A
コンパイル要求管理部
16A
キュー記憶部
15A
キュー制御部
22
加算部
17
コンパイル要求取得部
18
メソッドコンパイル部
6
14
登録要求受付部
5
ネイティブコード記憶部
7
コンパイル済メソッド情報テーブル
3
12
メソッド種別判定部
13
ネイティブコード取出部
11
命令実行部
2
クラスファイル/クラスライブラリ記憶部
8
クラスロード部
9
命令取出部
10
命令解読部
命令実行処理部
21
20

図8

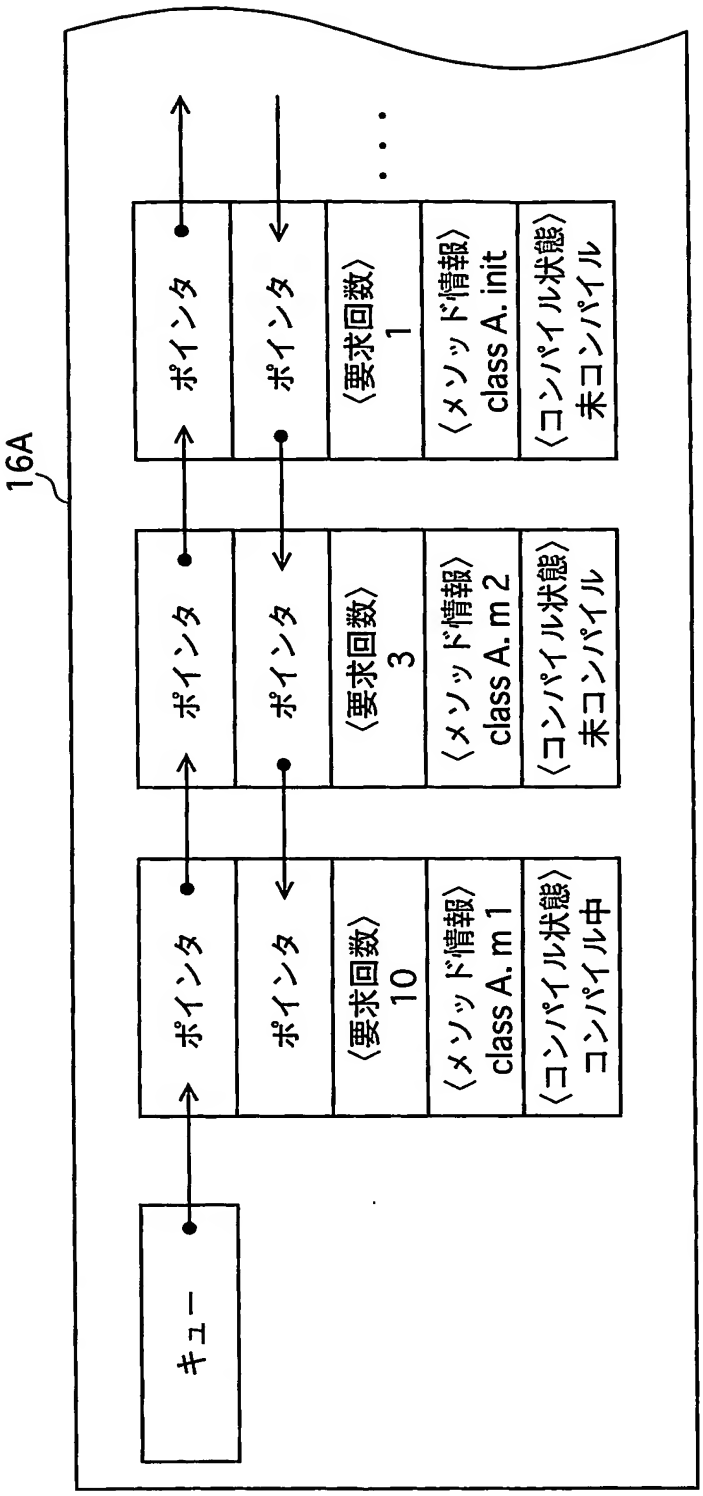

16A
キュー
ポインタ
ポインタ
〈要求回数〉
10
〈メソッド情報〉
class A. m 1
〈コンパイル状態〉
コンパイル中
ポインタ
ポインタ
〈要求回数〉
3
〈メソッド情報〉
class A. m 2
〈コンパイル状態〉
未コンパイル
ポインタ
ポインタ
〈要求回数〉
1
〈メソッド情報〉
class A. init
〈コンパイル状態〉
未コンパイル
・・・

図9

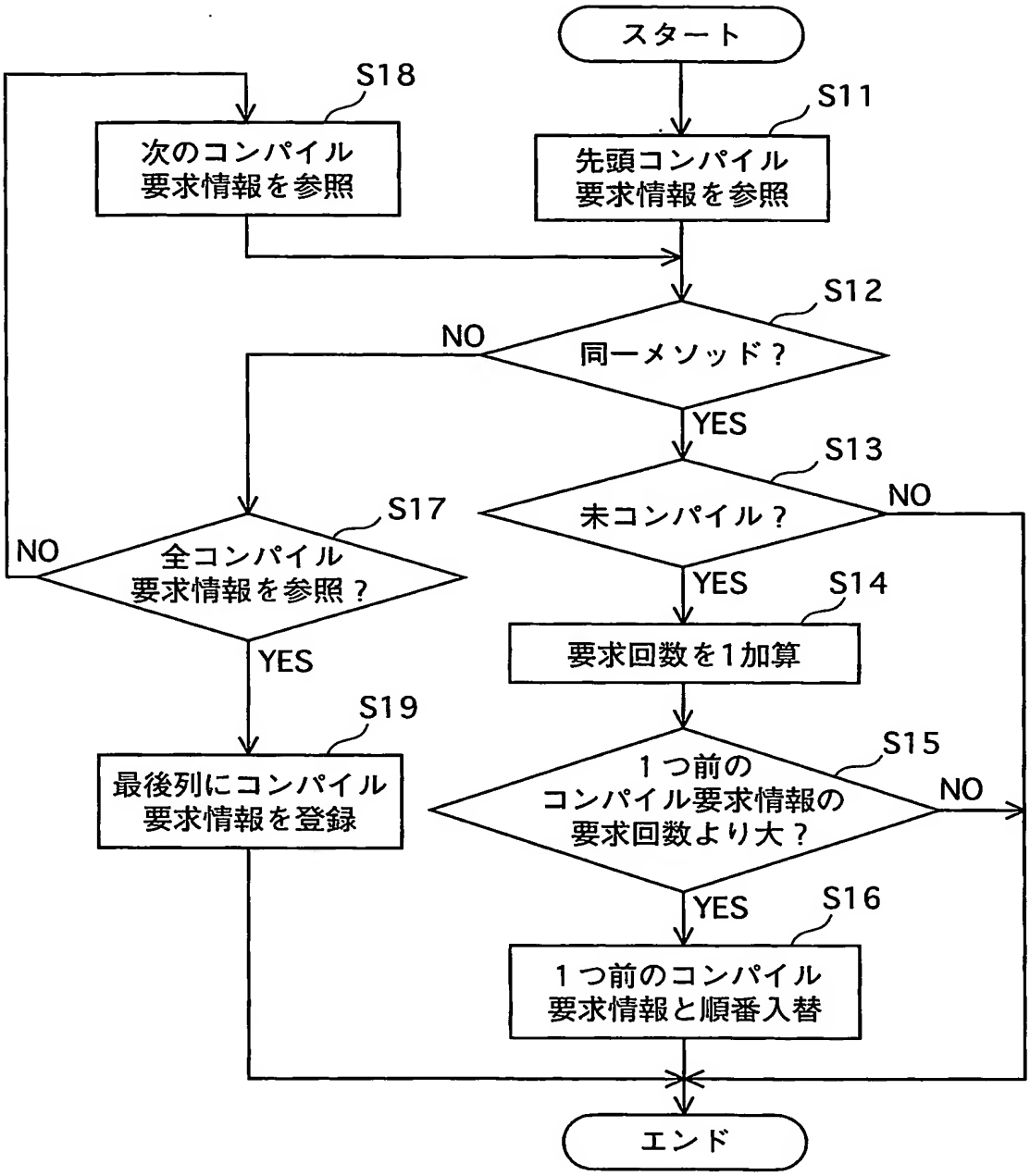

スタート
S11
先頭コンパイル
要求情報を参照
S12
同一メソッド？
NO
YES
S13
未コンパイル？
NO
YES
S14
要求回数を1加算
S15
1つ前の
コンパイル要求情報の
要求回数より大？
NO
YES
S16
1つ前のコンパイル
要求情報と順番入替
S17
全コンパイル
要求情報を参照？
NO
YES
S18
次のコンパイル
要求情報を参照
S19
最後列にコンパイル
要求情報を登録
エンド

図10

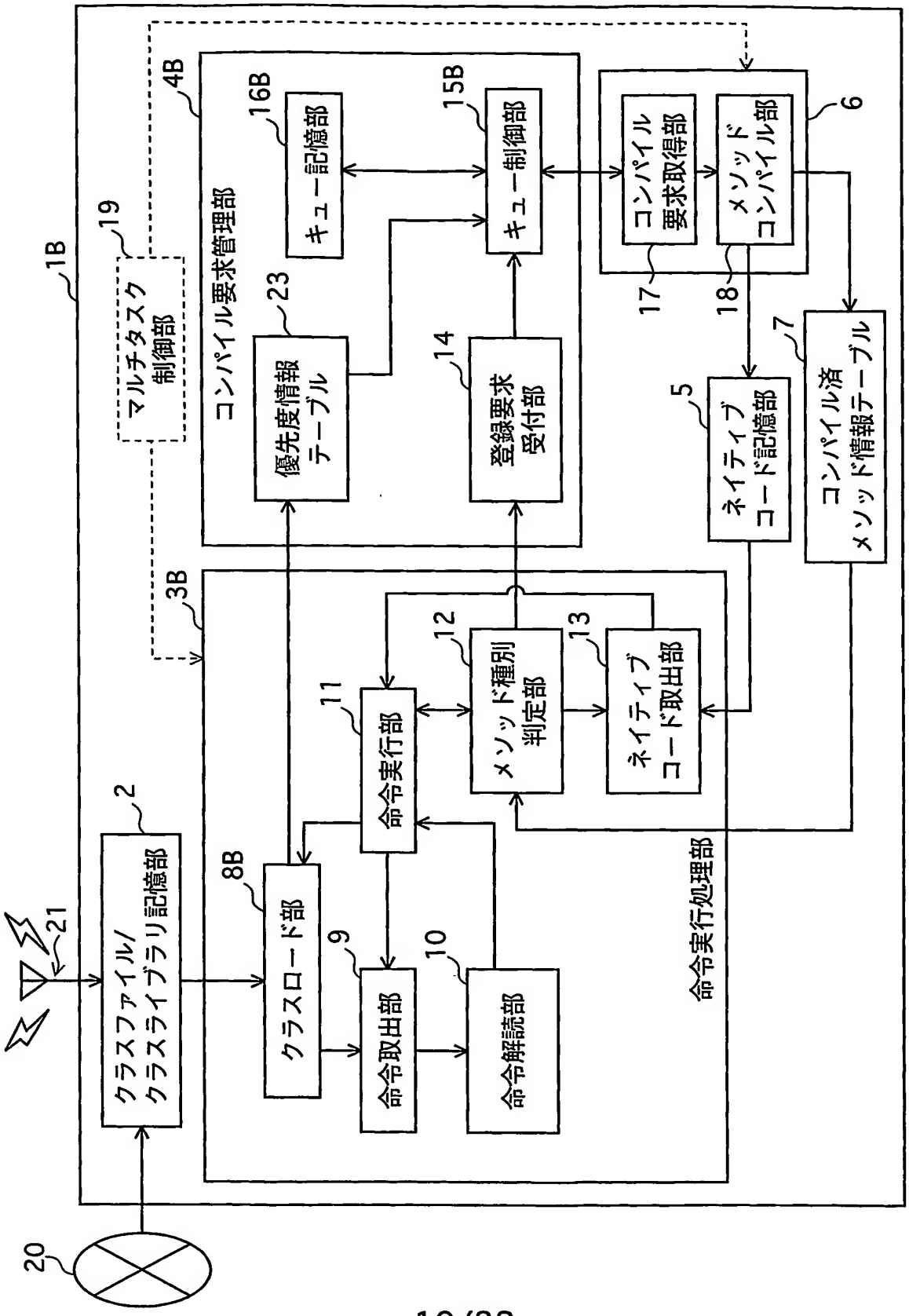

1B
19
マルチタスク制御部
4B
コンパイル要求管理部
16B
キュー記憶部
15B
キュー制御部
23
優先度情報テーブル
14
登録要求受付部
6
コンパイル要求取得部
メソッドコンパイル部
17
18
5
ネイティブコード記憶部
7
コンパイル済メソッド情報テーブル
3B
12
メソッド種別判定部
13
ネイティブコード取出部
11
命令実行部
2
クラスファイル/クラスライブラリ記憶部
8B
クラスロード部
9
命令取出部
10
命令解読部
命令実行処理部
21
20

図11

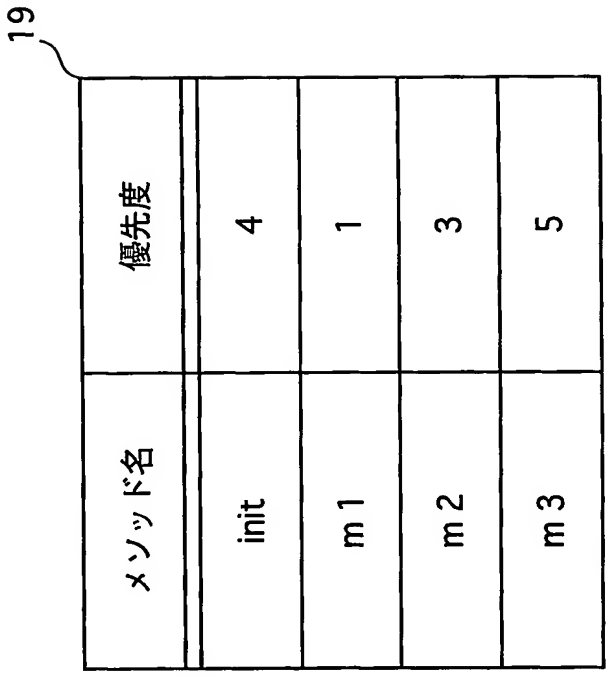


図12

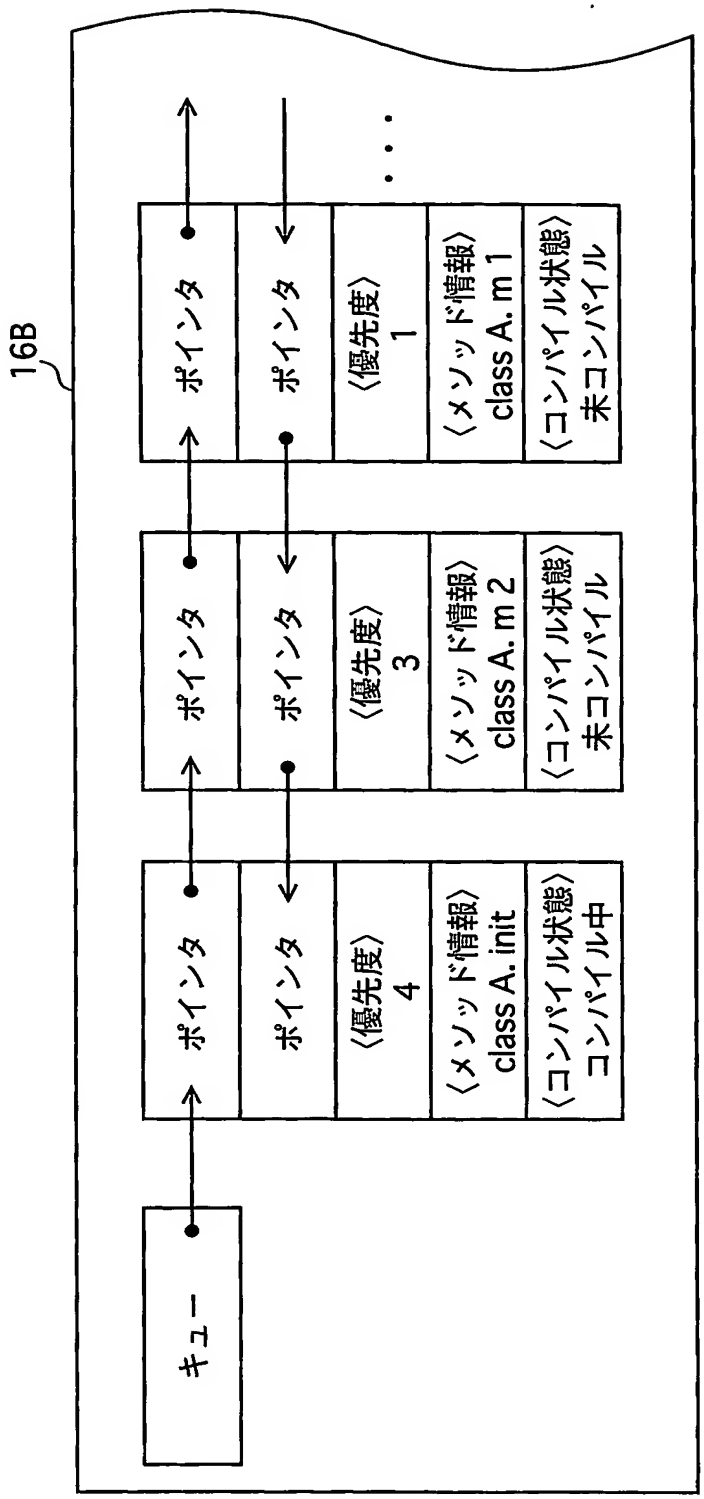

16B
キュー
ポインタ
ポインタ
〈優先度〉
4
〈メソッド情報〉
class A. init
〈コンパイル状態〉
コンパイル中
ポインタ
ポインタ
〈優先度〉
3
〈メソッド情報〉
class A. m 2
〈コンパイル状態〉
未コンパイル
ポインタ
ポインタ
〈優先度〉
1
〈メソッド情報〉
class A. m 1
〈コンパイル状態〉
未コンパイル

図13

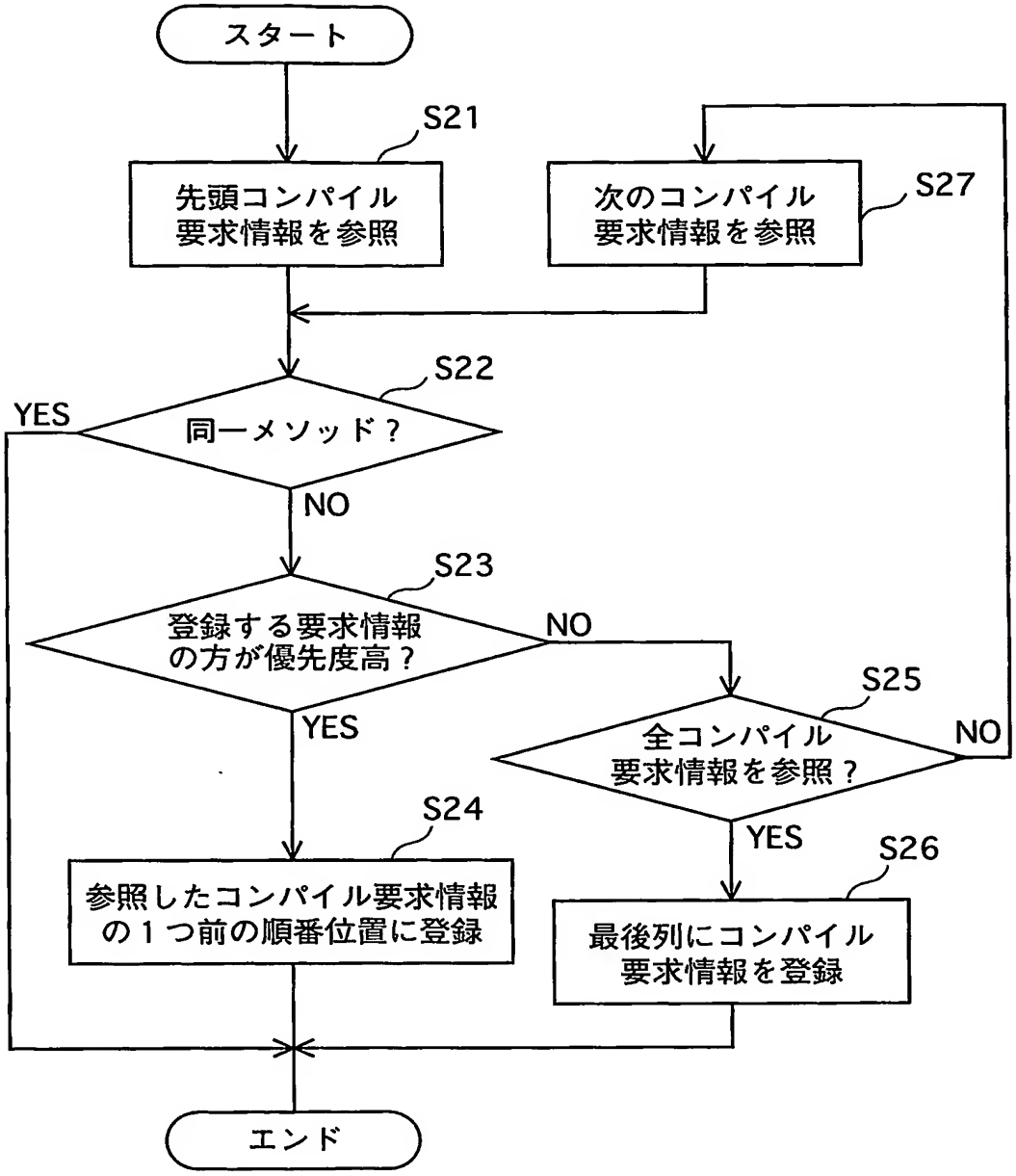

スタート
S21
先頭コンパイル
要求情報を参照
次のコンパイル
要求情報を参照
S27
S22
YES
同一メソッド？
NO
S23
登録する要求情報
の方が優先度高？
NO
YES
S25
全コンパイル
要求情報を参照？
NO
YES
S24
参照したコンパイル要求情報
の１つ前の順番位置に登録
S26
最後列にコンパイル
要求情報を登録
エンド

図14

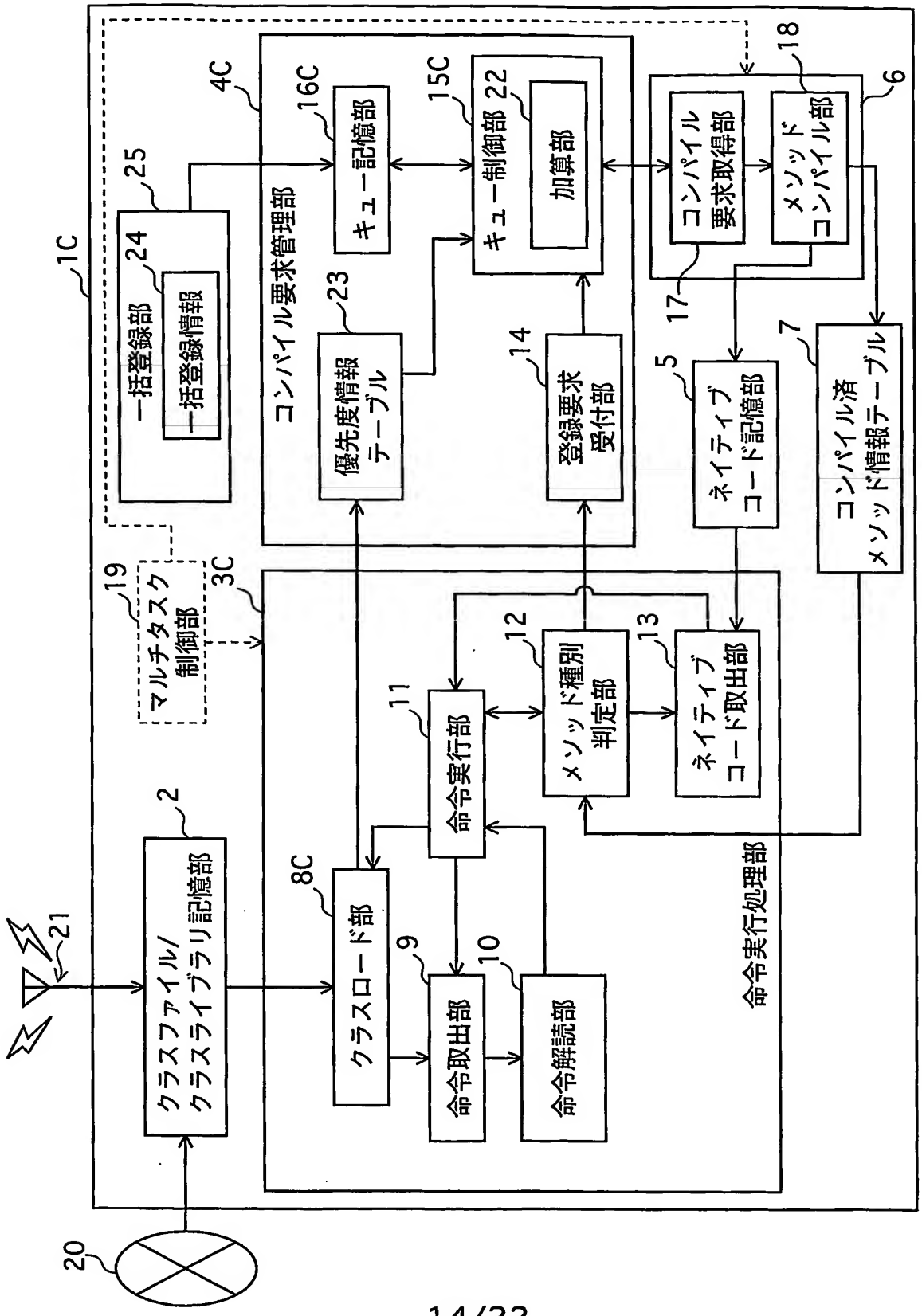

1C
2
クラスファイル/
クラスライブラリ記憶部
21
20
3C
マルチタスク
制御部
19
4C
コンパイル要求管理部
一括登録部
24
一括登録情報
25
キュー記憶部
16C
優先度情報
テーブル
23
キュー制御部
15C
加算部
22
登録要求
受付部
14
コンパイル
要求取得部
メソッド
コンパイル部
18
6
17
ネイティブ
コード記憶部
5
コンパイル済
メソッド情報テーブル
7
命令実行処理部
8C
クラスロード部
9
命令取出部
10
命令解読部
11
命令実行部
メソッド種別
判定部
12
ネイティブ
コード取出部
13

図15

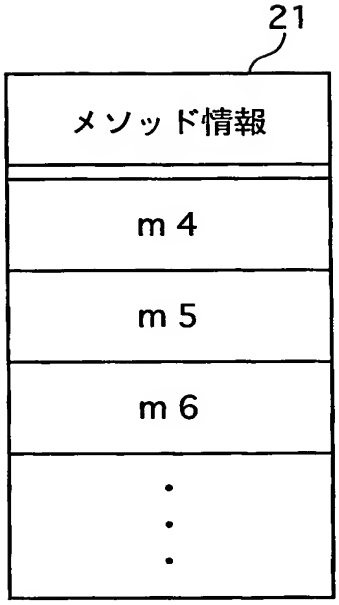


図16

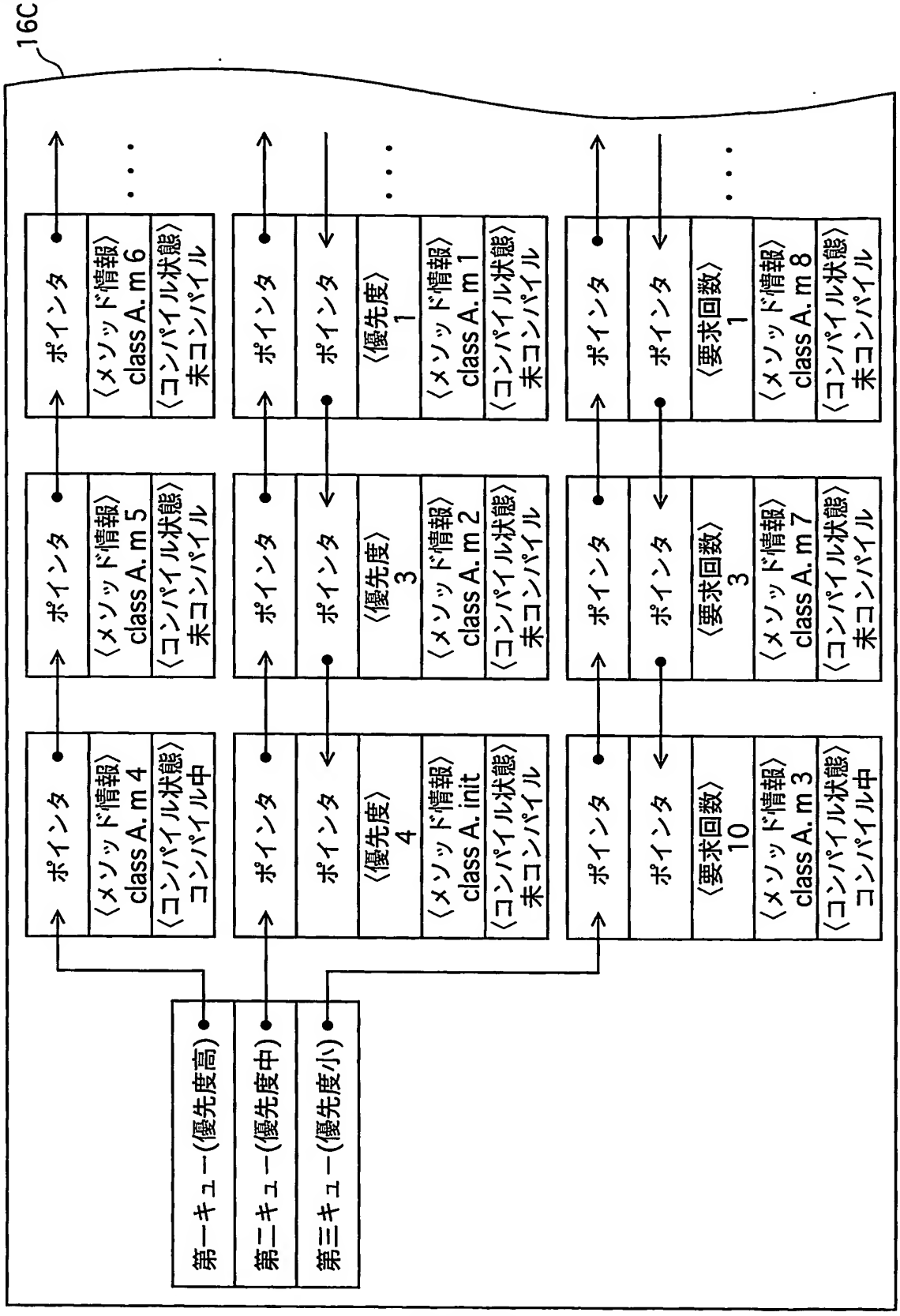

16C
第一キュー(優先度高)
第二キュー(優先度中)
第三キュー(優先度小)
ポインタ
〈メソッド情報〉 class A. m 4
〈コンパイル状態〉 コンパイル中
ポインタ
〈メソッド情報〉 class A. m 5
〈コンパイル状態〉 未コンパイル
ポインタ
〈メソッド情報〉 class A. m 6
〈コンパイル状態〉 未コンパイル
ポインタ
ポインタ
〈優先度〉 4
〈メソッド情報〉 class A. init
〈コンパイル状態〉 未コンパイル
ポインタ
ポインタ
〈優先度〉 3
〈メソッド情報〉 class A. m 2
〈コンパイル状態〉 未コンパイル
ポインタ
ポインタ
〈優先度〉 1
〈メソッド情報〉 class A. m 1
〈コンパイル状態〉 未コンパイル
ポインタ
ポインタ
〈要求回数〉 10
〈メソッド情報〉 class A. m 3
〈コンパイル状態〉 コンパイル中
ポインタ
ポインタ
〈要求回数〉 3
〈メソッド情報〉 class A. m 7
〈コンパイル状態〉 未コンパイル
ポインタ
ポインタ
〈要求回数〉 1
〈メソッド情報〉 class A. m 8
〈コンパイル状態〉 未コンパイル

図17

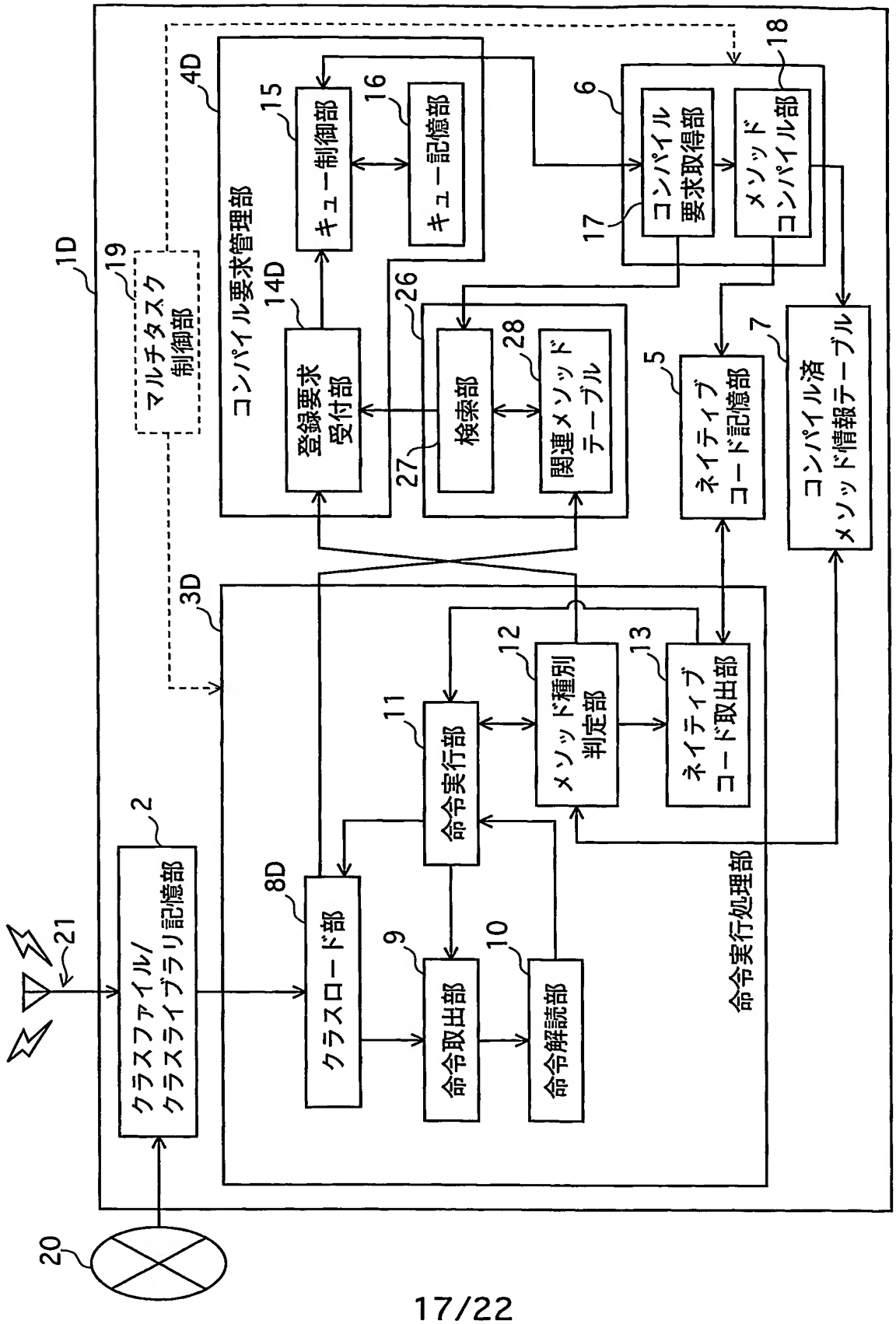

1D
19
マルチタスク制御部
4D
コンパイル要求管理部
15
キュー制御部
16
キュー記憶部
14D
登録要求受付部
26
検索部
28
関連メソッドテーブル
27
6
コンパイル要求取得部
17
メソッドコンパイル部
18
5
ネイティブコード記憶部
7
コンパイル済メソッド情報テーブル
3D
12
メソッド種別判定部
13
ネイティブコード取出部
11
命令実行部
命令実行処理部
2
クラスファイル/クラスライブラリ記憶部
8D
クラスロード部
9
命令取出部
10
命令解読部
21
20

図18

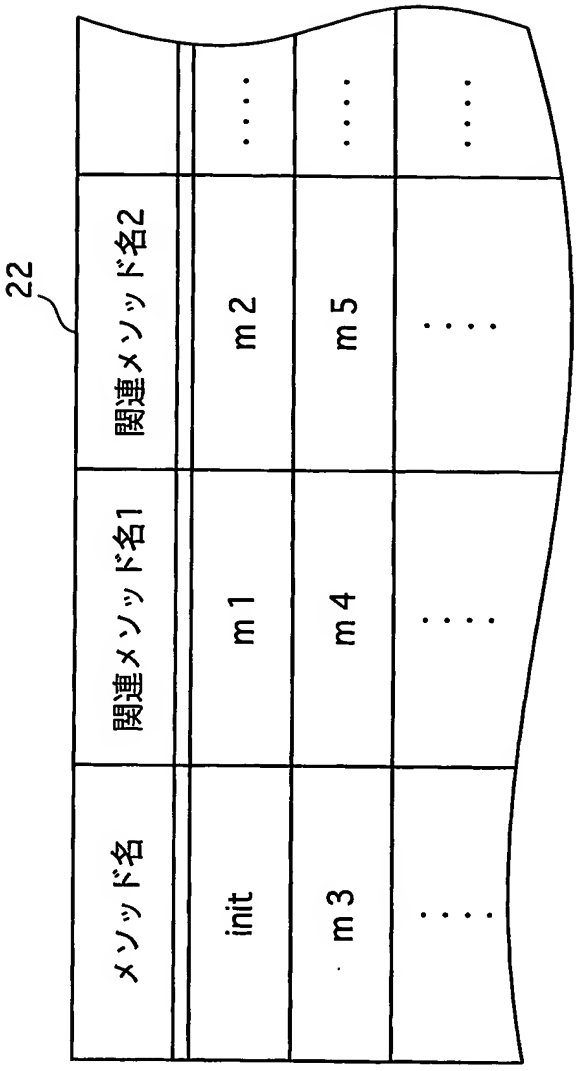


図19

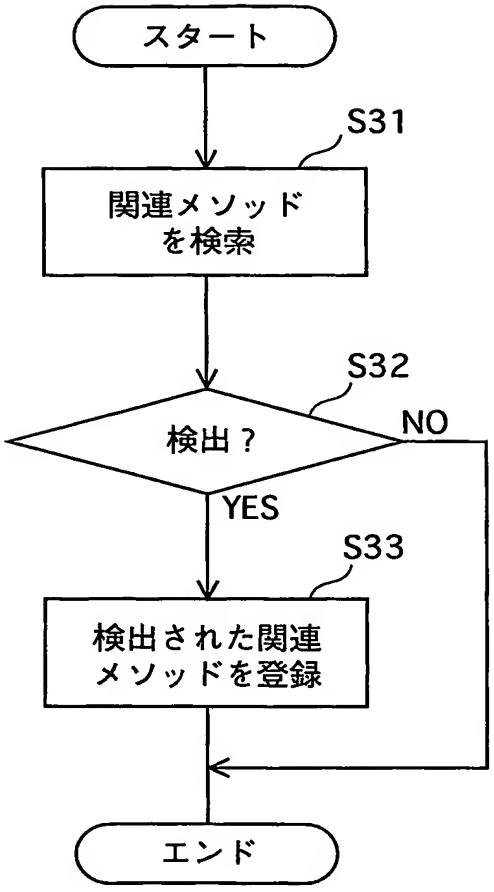


図20

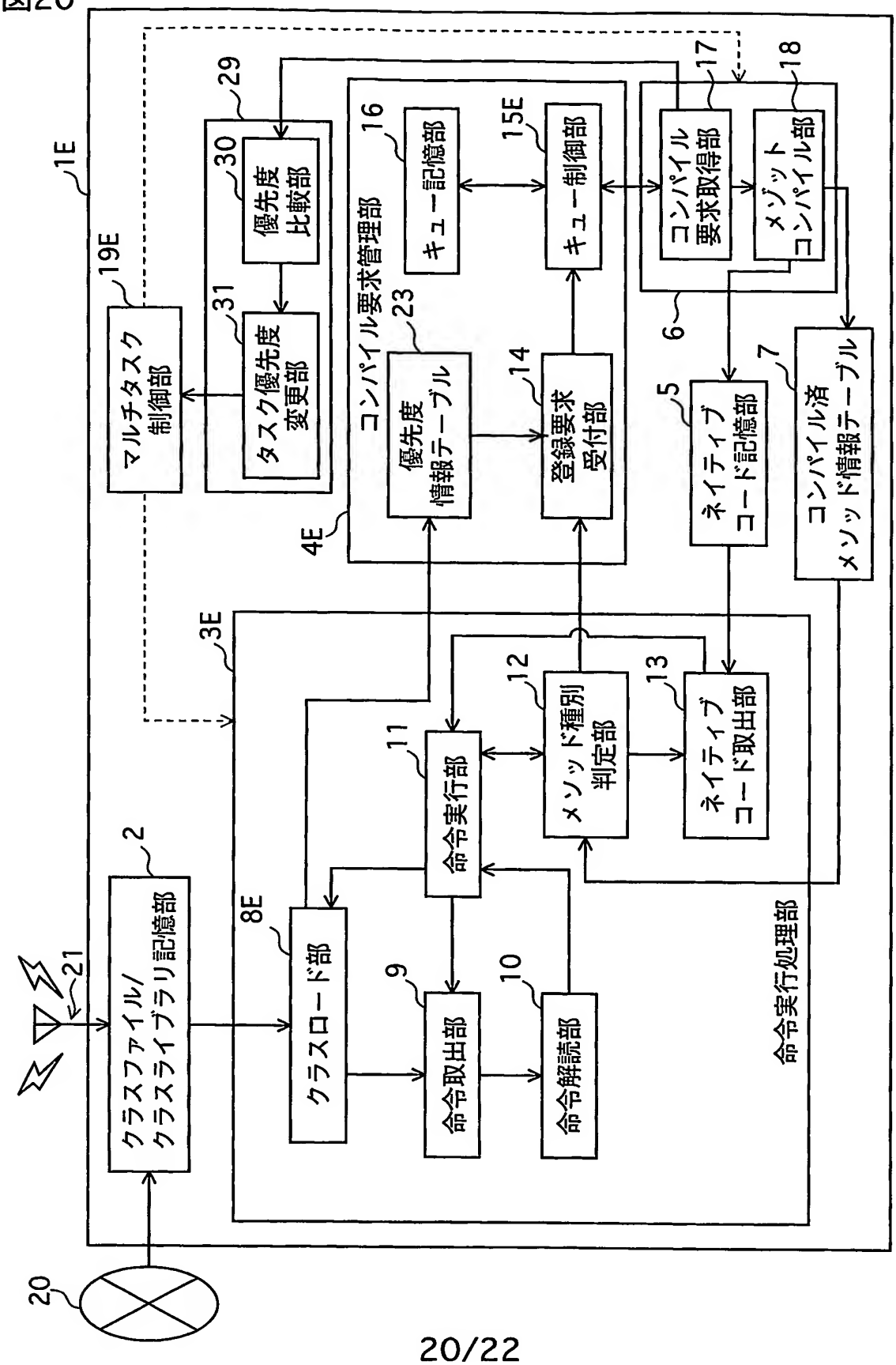

1E
2
クラスファイル/
クラスライブラリ記憶部
21
20
3E
8E
クラスロード部
9
命令取出部
10
命令解読部
11
命令実行部
12
メソッド種別
判定部
13
ネイティブ
コード取出部
命令実行処理部
4E
コンパイル要求管理部
16
キュー記憶部
15E
キュー制御部
17
コンパイル
要求取得部
18
メゾッド
コンパイル部
6
5
ネイティブ
コード記憶部
7
コンパイル済
メソッド情報テーブル
23
優先度
情報テーブル
14
登録要求
受付部
29
30
優先度
比較部
31
タスク優先度
変更部
19E
マルチタスク
制御部

図21

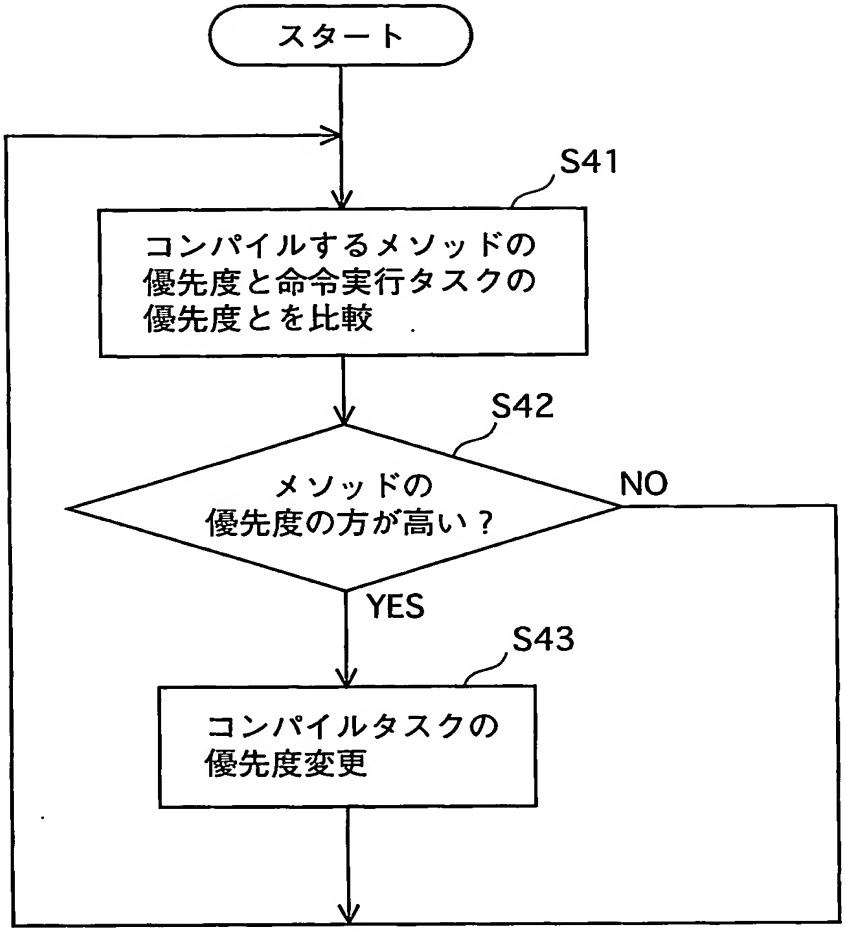
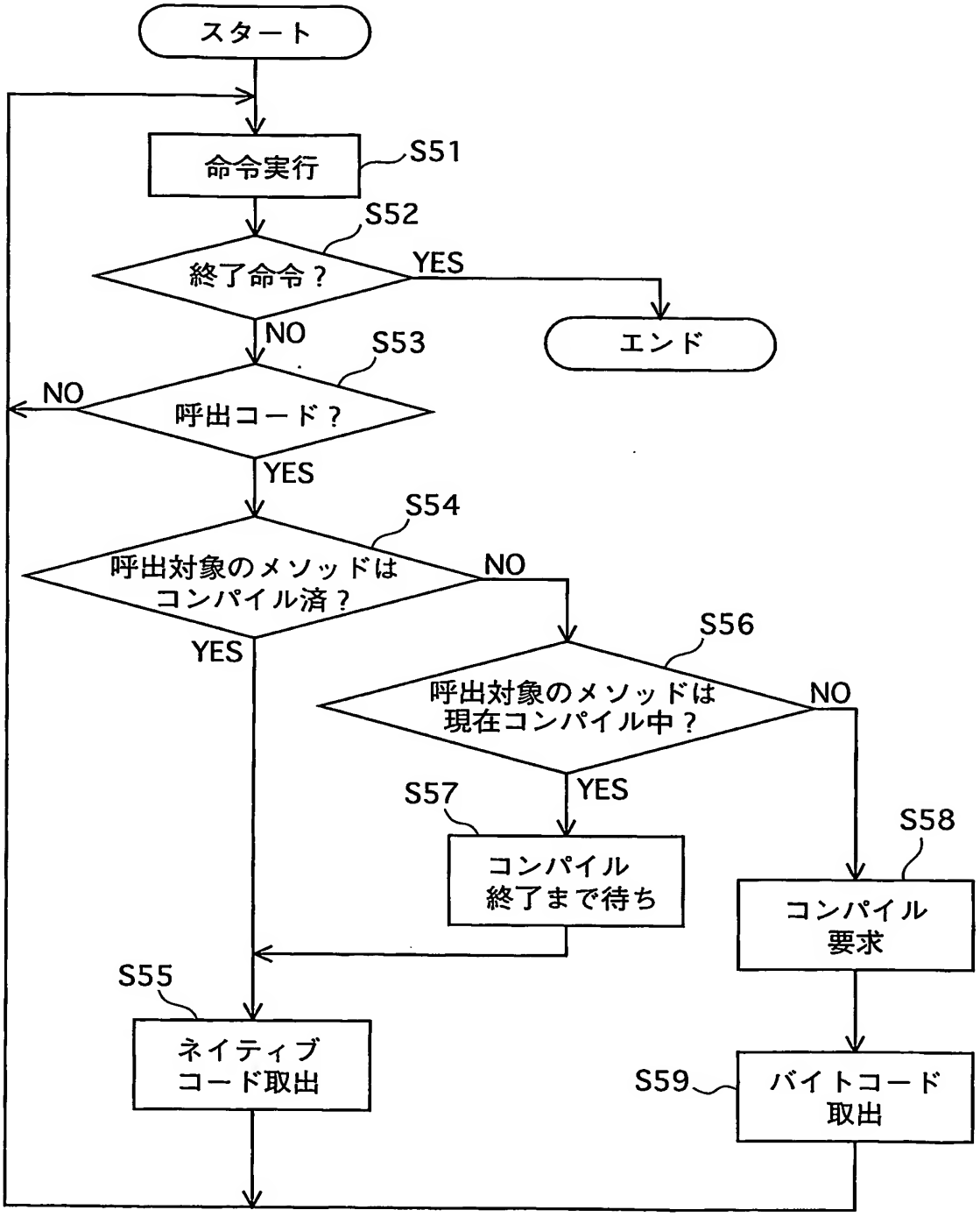

スタート
S41
コンパイルするメソッドの
優先度と命令実行タスクの
優先度とを比較
S42
メソッドの
優先度の方が高い？
NO
YES
S43
コンパイルタスクの
優先度変更

図22


スタート
命令実行
S51
S52
終了命令？
YES
エンド
NO
S53
NO
呼出コード？
YES
S54
呼出対象のメソッドは
コンパイル済？
NO
YES
S56
呼出対象のメソッドは
現在コンパイル中？
NO
YES
S57
コンパイル
終了まで待ち
S58
コンパイル
要求
S55
ネイティブ
コード取出
S59
バイトコード
取出

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/007731

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G06F9/45

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G06F9/45

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2004
Kokai Jitsuyo Shinan Koho 1971-2004 Toroku Jitsuyo Shinan Koho 1994-2004

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y A	JP 11-237989 A (Sun Micro Systems Inc.), 31 August, 1999 (31.08.99), Full text; all drawings & US 5970249 A & EP 908818 A2	1-7,9-13,15, 17-20 8 14,16
X,P Y,P A,P	JP 2004-118367 A (Mitsubishi Electric Corp.), 15 April, 2004 (15.04.04), Par. Nos. [0048] to [0067] (Family: none)	1-7,9-13,15, 17-20 8 14,16
Y	JP 2002-163115 A (Toshiba Corp.), 07 June, 2002 (07.06.02), Par. Nos. [0059] to [0066] (Family: none)	8

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search
25 August, 2004 (25.08.04)

Date of mailing of the international search report
14 September, 2004 (14.09.04)

Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

Form PCT/ISA/210 (second sheet) (January 2004)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/007731

Box No. II **Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)**

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III **Observations where unity of invention is lacking (Continuation of item 3 of first sheet)**

This International Searching Authority found multiple inventions in this international application, as follows:

The inventions of claims 1-5, 7, 9-13, 15, 17-20 has no "special technical feature" since they are identical to the invention disclosed in JP 11-237989 A. Accordingly,

the inventions of claims 1-5, 7, 9-13, 15, 17-20 relate to performing conversion processing in parallel to execution.

The invention of claim 6 relates to not performing dual registration.

The invention of claim 8 relates to the associated byte code string.

The invention of claim 14 relates to judgment whether the conversion processing is in progress.

(Continued to extra sheet.)

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☒ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest.

☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/007731

Continuation of Box No.III of continuation of first sheet(2)

The invention of claim 16 relates to making the priority of compile task higher.

国際調査報告

国際出願番号　PCT／JP2004／007731

A.　発明の属する分野の分類（国際特許分類（IPC））
Int. Cl.' G06F9/45

B.　調査を行った分野
調査を行った最小限資料（国際特許分類（IPC））
Int. Cl.' G06F9/45

最小限資料以外の資料で調査を行った分野に含まれるもの
日本国実用新案公報　1922－1996年
日本国公開実用新案公報　1971－2004年
日本国実用新案登録公報　1996－2004年
日本国登録実用新案公報　1994－2004年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
X Y A	JP　11－237989　A（サン・マイクロシステムズ・インコーポレイテッド）1999．08．31，全文，全図 & US　5970249　A & EP　908818　A2	1-7, 9-13, 15, 17-20 8 14, 16
X, P Y, P A, P	JP　2004－118367　A（三菱電機株式会社）2004．04．15，段落【0048】－【0067】 （ファミリーなし）	1-7, 9-13, 15, 17-20 8 14, 16

☒ C欄の続きにも文献が列挙されている。　☐ パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「A」特に関連のある文献ではなく、一般的技術水準を示すもの
「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「O」口頭による開示、使用、展示等に言及する文献
「P」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「&」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
25．08．2004	14. 9. 2004

国際調査機関の名称及びあて先	特許庁審査官（権限のある職員）		
日本国特許庁（ISA／JP） 郵便番号100－8915 東京都千代田区霞が関三丁目4番3号	中野　裕二	5B	9462
	電話番号　03－3581－1101　内線　3545		

様式PCT／ISA／210（第2ページ）（2004年1月）

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
Ｙ	ＪＰ　２００２－１６３１１５　Ａ（株式会社東芝） ２００２．０６．０７，段落【0059】－【0066】 （ファミリーなし）	8

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００４／００７７３１

第Ⅱ欄　請求の範囲の一部の調査ができないときの意見（第１ページの２の続き）

法第８条第３項（ＰＣＴ１７条(2)(a)）の規定により、この国際調査報告は次の理由により請求の範囲の一部について作成しなかった。

1．☐ 請求の範囲 _____ は、この国際調査機関が調査をすることを要しない対象に係るものである。
つまり、

2．☐ 請求の範囲 _____ は、有意義な国際調査をすることができる程度まで所定の要件を満たしていない国際出願の部分に係るものである。つまり、

3．☐ 請求の範囲 _____ は、従属請求の範囲であってＰＣＴ規則6.4(a)の第２文及び第３文の規定に従って記載されていない。

第Ⅲ欄　発明の単一性が欠如しているときの意見（第１ページの３の続き）

次に述べるようにこの国際出願に二以上の発明があるとこの国際調査機関は認めた。

請求の範囲１－５、７、９－１３、１５、１７－２０に係る発明は、特開平１１－２３７９８９号公報記載の発明と同一であるため、「特別な技術的特徴」を有しない。したがって、

請求の範囲１－５、７、９－１３、１５、１７－２０は、実行と並行して変換処理をすることに関するものである。

請求の範囲６は、二重登録を行わないことに関するものである。

請求の範囲８は、関連バイトコード列に関するものである。

請求の範囲１４は、変換処理が途中であるか判定することに関するものである。

請求の範囲１６は、コンパイルタスクの優先度を高くすることに関するものである。

1．☐ 出願人が必要な追加調査手数料をすべて期間内に納付したので、この国際調査報告は、すべての調査可能な請求の範囲について作成した。

2．☒ 追加調査手数料を要求するまでもなく、すべての調査可能な請求の範囲について調査することができたので、追加調査手数料の納付を求めなかった。

3．☐ 出願人が必要な追加調査手数料を一部のみしか期間内に納付しなかったので、この国際調査報告は、手数料の納付のあった次の請求の範囲のみについて作成した。

4．☐ 出願人が必要な追加調査手数料を期間内に納付しなかったので、この国際調査報告は、請求の範囲の最初に記載されている発明に係る次の請求の範囲について作成した。

追加調査手数料の異議の申立てに関する注意

☐ 追加調査手数料の納付と共に出願人から異議申立てがあった。

☐ 追加調査手数料の納付と共に出願人から異議申立てがなかった。